

Programa de Pós-Graduação em Engenharia Elétrica - PPGEL

Associação ampla UFSJ / CEFET-MG

Universidade Federal de São João del-Rei - UFSJ

Método Intervalar para Simulação de Mapas Discretos

Heitor Magno Rodrigues Junior

Orientador : Prof. Dr. Erivelton Geraldo Nepomuceno

São João del-Rei, 15 de dezembro de 2016.



Programa de Pós-Graduação em Engenharia Elétrica - PPGEL

Associação ampla UFSJ / CEFET-MG

Universidade Federal de São João del-Rei - UFMG

Método Intervalar para Simulação de Mapas Discretos

Heitor Magno Rodrigues Junior

Dissertação apresentada à banca examinadora designada pelo Colegiado do Programa de Pós-Graduação em Engenharia Elétrica, associação ampla entre a Universidade Federal de São João del-Rei e o Centro Federal de Educação Tecnológica de Minas Gerais, como parte dos requisitos necessários à obtenção do grau de Mestre em Engenharia Elétrica.

Orientador : Prof. Dr. Erivelton Geraldo Nepomuceno

São João del-Rei, 15 de dezembro de 2016.

Ficha catalográfica elaborada pela Divisão de Biblioteca (DIBIB)
e Núcleo de Tecnologia da Informação (NTINF) da UFSJ,
com os dados fornecidos pelo(a) autor(a)

R94m

Rodrigues Junior, Heitor Magno.
Método Intervalar para Simulação de Mapas Discretos / Heitor Magno Rodrigues Junior ; orientador Erivelton Geraldo Nepomuceno. -- São João del-Rei, 2016.
83 p.

Dissertação (Mestrado - Engenharia Elétrica) -- Universidade Federal de São João del-Rei, 2016.

1. Análise Intervalar. 2. Pseudo-Órbitas. 3. Mapa Logístico. 4. Órbitas Periódicas. I. Nepomuceno, Erivelton Geraldo, orient. II. Título.

Dedico este trabalho à minha família, meus professores e minha namorada pelo incentivo e apoio ao longo de minha vida acadêmica.

Agradecimentos

Agradeço a todos que participaram de forma direta ou indireta da conclusão dessa etapa.

Aos meus pais, pelo apoio, confiança e investimento na minha formação.

À Jéssica, por estar ao meu lado durante todos esses anos.

Aos demais familiares, que sempre me incentivaram.

À Universidade Federal de São João del-Rei (UFSJ) pela oportunidade de estudos e utilização de suas instalações.

Ao Professor Erivelton Nepomuceno, pela orientação, apoio e por todo conhecimento passado ao longo dos últimos anos.

Aos demais professores da UFSJ pelo conhecimento compartilhado.

Aos amigos, pelo companheirismo.

À Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - CAPES pelo apoio financeiro.

Resumo

Resultados obtidos por cálculos numéricos nem sempre são totalmente precisos. Isso ocorre devido ao fato de o computador apresentar limitações, fazendo com que o conjunto dos números que ele represente seja finito. Com o intuito de reduzir ou eliminar os erros causados por essas limitações, diversos métodos foram desenvolvidos ao longo dos anos. A análise intervalar é um método que procura limitar os erros computacionais, de modo que um resultado obtido seja representado por uma faixa de valores. Neste trabalho, uma metodologia de análise intervalar é implementada com o intuito de produzir métodos de detecção de pontos fixos e órbitas periódicas em mapas discretos. Os resultados obtidos com a metodologia desenvolvida nesta dissertação serão avaliados em relação a resultados de métodos presentes na literatura. Além disso, eles serão comparados com outros métodos de análise intervalar de acordo com a resposta atingida por cada um, onde serão avaliados tanto a precisão dos resultados quanto o custo computacional necessário para alcançá-los. A metodologia do trabalho se divide em dois métodos intervalares, onde o primeiro realiza operações aritméticas computacionais com os limites do intervalo ao invés de operações intervalares convencionais. O segundo método é baseado na propriedade da interseção intervalar e nele se faz a tentativa de reduzir o tamanho dos intervalos obtidos como resposta. Ambos os métodos são avaliados e comparados com outros métodos intervalares em 12 cálculos de ponto fixo e órbitas periódicas do mapa logístico. Aplicando a metodologia desenvolvida no trabalho, foi possível obter órbitas periódicas de período até 2048 com nível alto de confiabilidade na resposta, o que não foi possível nos demais métodos intervalares usados para comparação.

Palavras-chave: Análise Intervalar, Pseudo-Órbitas, Mapa Logístico, Órbitas Periódicas.

Abstract

The results obtained by numerical computation are not always totally precise. It happens because a computer has storage limitations and its set of numbers is finite. In order to reduce or eliminate errors caused by such limitations, various methods have been developed over the years. The interval analysis is a method that limits the computational errors. Thus an obtained result is represented by a range of values. In this work a methodology of interval analysis is implemented in order to generate detection methods of fixed points and periodic orbits in discrete maps. The results obtained with the methodology developed in this work will be assessed in relation to results in the literature. Moreover, the results will be compared with other interval methods according to the response obtained by each, which will be evaluated both the accuracy of the results and the computational cost required to achieve them. The methodology of this work is divided into two interval methods. The first performs computational arithmetic operations with the limits of the intervals instead of interval conventional operations. The second method is based on properties of interval analysis and makes an attempt to reduce the size of the intervals obtained as result. Both the first and the junction of the two methods are evaluated and compared with other interval methods in 12 cases of fixed point and periodic orbits calculations of the logistic map. Applying the methodology developed at work, it was possible to obtain periodic orbits with period up to 2048 with high precision, which was not possible in other methods analyzed.

Keywords: Interval Analysis, Pseudo-Orbits, Logistic Map, Periodic Orbits.

Lista de Figuras

1.1 Simulação em Matlab da Eq. (1.1) com $r = 327/100$ e $x_0 = 100/327$. 2

1.2 Simulação em Matlab para duas funções matematicamente equivalentes a (1.1). 3

2.1 Simulações do mapa senoidal para o cálculo do limite inferior de erro. 15

3.1 Tamanho, ponto médio e limites de um intervalo X. 26

3.2 Diagrama de bifurcação do mapa logístico. 40

3.3 *Cobweb plot* do mapa logístico com $r = 4$ e $x_0 = 2$. 41

5.1 Evolução de dois intervalos com o mesmo tamanho no mapa logístico. 62

5.2 Tamanhos de F_1, F_2 e $F_1 \cap F_2$ no Caso 7. 63

Lista de Tabelas

2.1 Simulação de Nepomuceno (2014) para o mapa logístico. 13

3.1 Disposição dos bits segundo padrão IEEE 754-2008 para sistemas de 32 e 64 bits. 19

4.1 As 10 primeiras iterações do mapa logístico com $r = 3{,}6$ e $x_0 = 0{,}6$. 48

4.2 Pontos de duplicação de período para o mapa logístico. 53

5.1 Comparação dos cálculos para o Caso 1 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 55

5.2 Comparação dos cálculos para o Caso 2 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 56

5.3 Comparação dos cálculos para o Caso 3 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 56

5.4 Comparação dos cálculos para o Caso 4 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 57

5.5 Comparação dos cálculos para o Caso 5 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 57

5.6 Comparação dos cálculos para o Caso 6 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 57

5.7 Comparação dos cálculos para o Caso 7 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 58

5.8 Comparação dos cálculos para o Caso 8 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 58

5.9 Comparação dos cálculos para o Caso 9 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 59
5.10 Comparação dos cálculos para o Caso 10 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 59
5.11 Comparação dos cálculos para o Caso 11 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 59
5.12 Comparação dos cálculos para o Caso 12 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014). 59
5.13 Média de tamanho dos intervalos ao longo dos cálculos e tamanho final dos intervalos obtidos com a metodologia desenvolvida para F_1, F_2 e $F_1 \cap F_2$. . . . 61

Lista de Símbolos

x_{n+1}	Função recursiva;
r	Parâmetro do mapa logístico;
n	Iteração do cálculo;
x_n	Saída obtida na iteração n para um mapa;
$\in$	É um elemento de;
$\mathbb{R}$	Conjunto dos números reais;
$\mathbb{N}$	Conjunto dos números naturais;
$\forall$	Para todo;
$\mathbb{D}$	Conjunto dos números que podem ser armazenado do computador;
$\subset$	Está contido em;
$\mathbb{Q}$	Conjunto dos números irracionais;
I	Intervalo;
J	Representação de um intervalo no computador;
f_n	Função iterativa;
$\hat{f}_n$	Representação de uma função iterativa no computador;
x	Número real;
$\hat{x}$	Representação de um número no computador;
$d()$	Distância;
$\leq$	Menor ou igual a;
$\subseteq$	É um subconjunto de;
$\delta_{\alpha,n}$	Limite Inferior do Erro;
$\hat{x}_{a,n}$	Elemento da pseudo-órbita de a, correspondente à iteração n;
$\epsilon_{\alpha,n}$	Precisão relativa;

(-x-)	Representação da curva com marcador em formato de x;
(-o-)	Representação da curva com marcador em formato de o;
$\pm$	Mais ou menos;
S	Significante ou mantissa;
$\times$	Indica multiplicação;
B	Base adotada;
E	Expoente;
b_0	Bit Escondido;
b_n	Bits do significante;
p	Precisão do sistema;
ϵ	Épsilon da máquina;
E_{min}	Expoente mínimo assegurado;
E_{max}	Expoente máximo assegurado;
N_{min}	Menor valor positivo normalizado representável pela máquina;
N_{max}	Maior valor positivo normalizado representável pela máquina;
$round(x)$	Arredondamento de um número x;
X	Representação de um intervalo;
$\underline{X}$	Limite inferior do intervalo X;
$\overline{X}$	Limite superior do intervalo X;
$\approx$	Aproximadamente;
$m(X)$	Ponto médio do intervalo X;
$\omega(X)$	Tamanho do intervalo X;
$\cap$	Interseção;
$\cup$	União;
$\min\{\}$	Valor mínimo dentro do conjunto;
$\max\{\}$	Valor máximo dentro do conjunto;
S	Conjunto com todas as possibilidades de multiplicação entre limites de dois intervalos;
$f(x)$	Função definida sob o conjunto dos números reais;
$f(X)$	Função intervalar;
$F(X)$	Extensão intervalar;

`intval`(x)	Função do *Intlab* que gera intervalos com arredondamentos de x;
`midrad`(x,rad)	Função do *Intlab* que define o ponto médio e o raio do intervalo;
`infsup`(i,s)	Função do *Intlab* que define um intervalo a partir dos limites;
rad	Raio do intervalo;
inf	Limite inferior do intervalo;
sup	Limite superior do intervalo;
$f^n(x_0)$	Função composta recursiva;
$x_{i,n}$	Número obtido por uma pseudo-órbita i na iteração n;
$\delta_{i,n}$	Diferença entre pseudo-órbita e a órbita verdadeira;
x^*	Ponto fixo;
r_∞	Valor do parâmetro r onde começa o regime caótico no mapa logístico;
$\rightarrow$	Implica;
$-\infty$	Menos infinito;
∞	Infinito;
δ	Distância entre um ponto limite e outro ponto qualquer do intervalo;
F_1	Cálculo realizado para a extensão intervalar F_1 em qualquer metodologia;
F_2	Cálculo realizado para a extensão intervalar F_1 em qualquer metodologia;
$F_1 \cap F_2$	Método que usa interseção de pseudo-órbitas;

Lista de Abreviações

UFSJ	Universidade Federal de São João del-Rei;
CEFET - MG	Centro Federal de Educação Tecnológica de Minas Gerais;
CAPES	Coordenação de Aperfeiçoamento de Pessoal de Nível Superior;
CNPq	Conselho Nacional de Desenvolvimento Científico e Tecnológico;
LECOM	Laboratório de Estudos em Controle e Modelagem;
DINCON	Conferência Brasileira de Dinâmica, Controle e Aplicações ;
CBA	Congresso Brasileiro de Automática;
IEEE	Institute of Electrical and Electronics Engineers;
IEEE 754-2008	Norma IEEE 754-2008 de Ponto Flutuante;
XSC	Linguagem de Computação Estendida;
RLC	Circuito elétrico composto por resistor, indutor e capacitor;
PASCAL-XSC	Ferramenta de computação estendida em Pascal;
C-XSC	Ferramenta de computação estendida na linguagem C;
PASCAL-XSC	Ferramenta de computação estendida na linguagem Fortran;
PASCAL-XSC	Ferramenta de computação estendida na linguagem Fortran;
FINCA	*Free Interval Circuit Analyzer*
[I]RMOEA	Algoritmo evolucionário intervalar para otimização robusta multiobjetivo;
PHWR	*Pressurized Heavy-Water Reactor*;
MWe	Megawatt elétrico;
PID	Controlador Proporcional Integral Derivativo;
IBM	Empresa *International Business Machines*;
ulp	unit in the last place;
NARMAX	*Nonlinear Autoregressive Moving Average Model with Exogenous* Inputs;

Sumário

Lista de Símbolos **xvii**

Lista de Símbolos **xxi**

1 Introdução **1**

1.1 Objetivos 4

1.2 Contribuições da Dissertação 5

1.3 Organização do Trabalho 6

2 Revisão de Literatura **9**

2.1 Trabalhos Relevantes 9

2.2 Análise de Erros em Funções Recursivas 12

2.3 Análise de Erros com o Polinômio NARMAX 13

2.4 Conclusões do Capítulo 15

3 Fundamentação Teórica **17**

3.1 Computação Numérica 17

3.1.1 Norma IEEE 754-2008 de Ponto Flutuante 17

3.1.2 Noções da Representação dos Números no Computador 18

3.1.3 Representação Computacional em Diferentes Formatos da Norma IEEE 754-2008 19

3.1.4 Precisão do Sistema 19

3.1.5 ULP (*Unit in the Last Place*) da Máquina 20

3.1.6 Arredondamento 21

3.1.7 Erros Computacionais 22
3.1.8 Comentários Finais da Seção 23
3.2 Análise Intervalar 24
3.2.1 Breve História da Análise Intervalar 24
3.2.2 Definições Básicas 25
3.2.3 Relações Algébricas de Intervalos 26
3.2.4 Operações Aritméticas Elementares 27
3.2.5 Propriedades Algébricas Intervalares 28
3.2.6 Funções Intervalares 31
3.2.7 Superestimação de Intervalos 31
3.2.8 Extensões Intervalares 32
3.2.9 Funções Racionais Intervalares 33
3.2.10 Ferramentas Computacionais Intervalares 34
3.2.11 Conclusões da Seção 35
3.3 Sistemas Dinâmicos Não-Lineares 36
3.3.1 Sistemas Dinâmicos Caóticos 36
3.3.2 Sistemas Dinâmicos em Tempo Discreto 37
3.3.3 Órbitas e Pseudo-Órbitas 38
3.3.4 Pontos Fixos e Órbitas Periódicas 38
3.4 Mapa Logístico 39
3.4.1 Diagrama de Bifurcação 39
3.4.2 *Cobweb Plot* 40
3.4.3 Comportamento do Mapa Logístico 41

4 Metodologia **43**
4.1 Trabalhando com os Pontos Limites 43
4.1.1 Arredondamento dos Resultados Obtidos 44
4.1.2 Aplicação do Método no Mapa Logístico 44
4.1.3 Analisando a Coerência do Método 47
4.1.4 Convergindo para Pontos Fixos e Órbitas Periódicas 48
4.2 Reduzindo o Intervalo das Soluções 49
4.3 Método Intervalar Desenvolvido 50

4.4 Propostas 52
4.5 *Hardware* e *Software* Utilizados 54

5 Resultados **55**

6 Conclusões **65**

6.1 Considerações Finais 65
6.2 Trabalhos Futuros 68

Referências Bibliográficas **69**

A Rotinas Computacionais **77**

A.1 Algoritmo da Aplicação de $F_1 \cap F_2$ 77
A.2 Algoritmo do Cálculo de F_1 e F_2 Sem Usar Interseção das Pseudo-Órbitas . . . 80

CAPÍTULO 1

Introdução

A computação numérica é parte vital da infraestrutura científica moderna, sendo utilizada para resolver equações que modelam os mais diferentes sistemas. Porém, apesar de serem ferramentas essenciais para o desenvolvimento científico, as técnicas de computação numérica apresentam resultados quase sempre inexatos. Esse fato ocorre devido à limitação de memória nas representações de números reais pela máquina (Kulisch, 1999).

Essa limitação vem da representação finita dos números reais, estabelecida e padronizada pela norma IEEE 754-2008 para aritmética de ponto flutuante (IEEE, 2008). Isso faz com que a aritmética computacional seja uma aproximação sistemática da aritmética dos números reais, sendo representada por um subconjunto finito dos números reais. Consequentemente, algumas propriedades da aritmética dos números reais não são garantidas para a aritmética de ponto flutuante (Overton, 2001).

Mesmo para dados de entrada que podem ser representados de forma exata pela máquina, o resultado de uma simples operação matemática com esse número pode não ser. Isso faz com que o computador apresente uma aproximação ao invés do resultado exato, gerando erro de arredondamento (Galias, 2013).

Um fato conhecido envolvendo falhas provocadas por erros de arredondamento foi o problema no *software* do sistema americano de defesa de mísseis aéreos *Patriot*, durante a Guerra do Golfo, em 1991. O algoritmo usado pelo radar do sistema de defesa levava em conta a velocidade do míssil atacante e o tempo. Como esses valores eram levemente truncados na conversão para aritmética de ponto flutuante, o erro acumulado ao fim de 100 horas de funcionamento era da ordem de aproximadamente 0,3433 segundos, correspondendo a um deslocamento na posição da região de procura do radar da ordem de 687 metros. Isso fez com que o sistema de

detecção de ataques inimigos falhasse, não conseguindo detectar um míssil lançado pela Arábia Saudita, causando a morte de 28 militares americanos e ferindo outros 98 (Marshall, 1992).

Na análise de soluções de sistemas dinâmicos não-lineares, a computação numérica também desempenha um papel crucial. No entanto, em sistemas mais complexos, a propagação de erros pode chegar a anular os resultados obtidos (Lozi, 2013).

Simples exemplos da influência da propagação do erro em cálculos numéricos de sistemas dinâmicos não-lineares representados por mapas discretos, podem ser mostrados em dois casos distintos na análise da sequência de pontos obtidos a cada iteração para a função do mapa logístico, dada pela função recursiva apresentada na equação (1.1).

$$x_{n+1} = r x_n (1 - x_n), \tag{1.1}$$

onde $r \in [0,4]$ e $x_n \in (0,1)$, $n \in \mathbb{N}$ (May, 1976).

No primeiro caso, mostrado em Nepomuceno (2014), a análise é feita através do *software Matlab* para $r = \frac{327}{100}$ com condição inicial $x_0 = \frac{100}{327}$. Nota-se na Figura 1.1 que após aproximadamente 150 iterações, a resposta apresenta convergência para dois valores distintos, que se repetem a cada duas iterações.

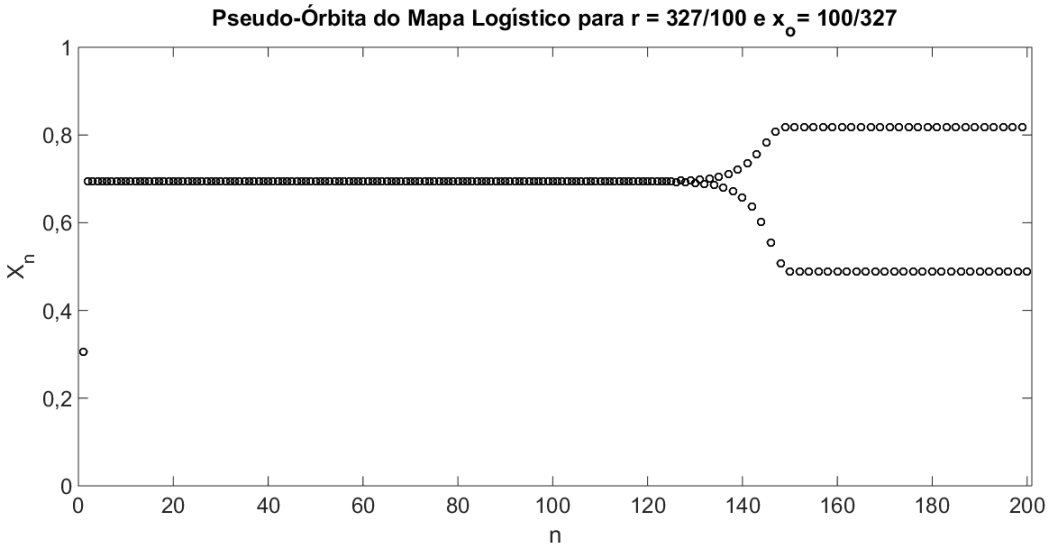


Figura 1.1: Simulação feita no Matlab R2014b da Eq. (1.1) com $r = 327/100$ e $x_0 = 100/327$, usando processador *Intel i7-5500 @ 2.41GHz*.

No entanto, analisando as primeiras iterações analiticamente através do conjunto de equações (1.2), claramente se percebe que a sequência das respostas obtidas deveria convergir para o ponto fixo $x_n = \frac{227}{327}$, $\forall\, n \in \mathbb{N}$.

$$x_1 = \frac{\cancel{327}}{\cancel{100}}.\frac{\cancel{100}}{\cancel{327}}.\left(1 - \frac{100}{327}\right) = \frac{327 - 100}{327} = \frac{227}{327},$$
$$x_2 = \frac{\cancel{327}}{100}.\frac{227}{\cancel{327}}.\left(1 - \frac{227}{327}\right) = \frac{227}{\cancel{100}}.\frac{\cancel{100}}{327} = \frac{227}{327}. \tag{1.2}$$

Observando o conjunto de equações (1.2), comprova-se que, por se tratar de uma função recursiva, onde a resposta da iteração atual é a entrada da próxima, tem-se que $x_1 = x_2 = \cdots = x_n = \frac{227}{327}$, invalidando o resultado obtido pelo computador.

O caso apresentado em Rodrigues Junior et al. (2016b) é outro exemplo de investigação do erro computacional, onde são apresentadas duas funções recursivas, dadas por $F(x_n) = rx_n(1 - x_n)$ e $G(x_n) = rx_n - rx_n^2$. A Figura 1.2 mostra a sequência de pontos obtidos com o *software Matlab* das iterações 50 a 70, para as funções $F(X_n)$ e $G(x_n)$ com $r = 3{,}9$ e $x_0 = 0{,}6$.

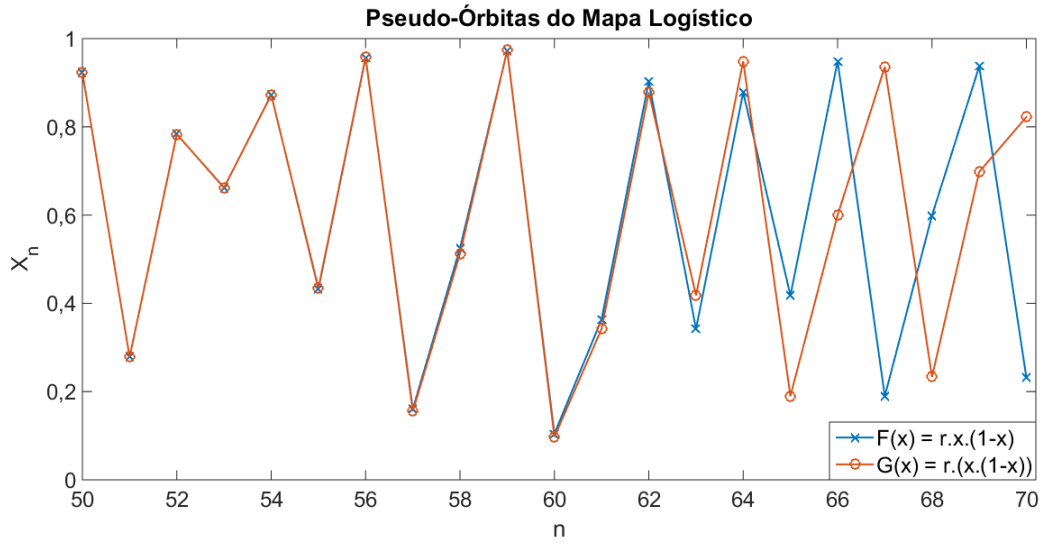


Figura 1.2: Simulação feita no Matlab R2014b da Eq. (1.1) com $r = 3{,}9$ e $x_0 = 0{,}6$ para duas funções matematicamente equivalentes, usando processador *Intel i7-5500 @ 2.41GHz*.

Por serem matematicamente equivalentes, para os mesmos parâmetros e condições iniciais, ambas funções deveriam apresentar as mesmas trajetórias de respostas. Porém, devido ao fato de apresentarem sequências distintas de operações aritméticas, seus resultados podem ser arredondados para valores diferentes. E com o passar das iterações, essa diferença pode ser propagada. Como se pode notar na Figura 1.2, essa diferença se torna evidente depois de aproximadamente 60 iterações.

Para se ter uma ideia da influência da propagação do erro neste caso, a distância entre as trajetórias na iteração $n = 67$ é maior do que $0{,}74$, ou seja, a diferença entre as respostas chega

a ser 74% do intervalo de variação da resposta. Isso mostra que apesar de serem duas funções matematicamente equivalentes, não se pode afirmar qual delas é a representação fiel da equação (1.1) no computador.

Segundo Ott (2002), muitos pesquisadores asseguram comportamentos caóticos de vários sistemas dinâmicos com confiança em soluções numéricas apresentadas a partir de softwares populares e computadores facilmente acessíveis para a maioria dos pesquisadores. No entanto, Lozi (2013) afirma que existem muitos trabalhos publicados cuja confiabilidade em seus resultados numéricos não é avaliada cuidadosamente.

Uma maneira de tentar reduzir, não só os problemas apresentados nos casos anteriores, mas também outros tipos de problemas que envolvem a confiabilidade de resultados em sistemas dinâmicos não-lineares representados por mapas discretos, é por meio da análise intervalar. Como o próprio nome mostra, é uma ferramenta que define conjuntos de intervalos ao invés de números reais, delimitando uma faixa de valores possíveis para representar uma dada informação (Hargreaves, 2002). Com isso, essa técnica permite usar um computador para obter resultados minuciosos, assegurando que as soluções obtidas contenham a solução verdadeira (Galias, 2002).

A aritmética intervalar vem sendo implementada em diversas plataformas e é suportada por diferentes linguagens de programação. As linguagens de computação científica estendida ou *extended scientific computation* (*XSC*) apresentam ferramentas poderosas e necessárias para se trabalhar com essa aritmética (Alefeld e Mayer, 2000). Dentre as diversas ferramentas de implementação de aritmética intervalar, o *Intlab*, *toolbox* do *Matlab*, se destaca pela facilidade e rapidez. Além desses fatores, é amplamente conhecido e abordado pela comunidade científica, como por exemplo em Hargreaves (2002), Mendonça e Campos (2013), Pereira e Da Costa (2014), Banerjee et al. (2015) e de Oliveira et al. (2016).

1.1 Objetivos

O principal objetivo do trabalho é desenvolver um método de análise intervalar no cálculo de pontos fixos e órbitas periódicas em mapas discretos. Além disso, são também objetivos a serem desenvolvidos nesta dissertação:

- Mostrar a eficácia do método para o mapa logístico (Eq. (1.1));

- Buscar a redução dos limites dos intervalos que contém a solução verdadeira;
- Desenvolver rotinas computacionais de fácil implementação com aplicação da computação por intervalos;
- Avaliar a metodologia desenvolvida através da comparação dos resultados com outros métodos e ferramentas intervalares;
- Utilizar a análise intervalar para confirmar dados encontrados na literatura.

1.2 Contribuições da Dissertação

Os principais trabalhos publicados em congressos ao longo do desenvolvimento desta dissertação são apresentados a seguir:

- Rodrigues Junior H. M. e Nepomuceno E. G. (2015), Uso da Computação Por Intervalos para Cálculo de Ponto Fixo de um Mapa Discreto. *Anais do DINCON 2015 - Conferência Brasileira de Dinâmica, Controle e Aplicações*, pp. 1–7 (Rodrigues Junior e Nepomuceno, 2015).
- Rodrigues Junior H. M., Peixoto M. L. C., Nardo L. G., Nazaré T. E. and Nepomuceno E. G. (2016), Cálculo de Pontos Fixos por Análise Intervalar para Mapas Discreto. *Anais do CBA 2016 - XXI Congresso Brasileiro de Automática*, pp, 1–6 (Rodrigues Junior et al., 2016a).
- Rodrigues Junior H. M., Peixoto M. L. C. and Nepomuceno E. G. (2016), Simulation of Recursive Functions by Means of Interval Analysis and Pseudo-Orbits. *Proceedings of 6th International Conference on Nonlinear Science and Complexity, NSC 2016*, pp. 1–4 (Rodrigues Junior et al., 2016b).

Em Rodrigues Junior e Nepomuceno (2015) é desenvolvido um método que leva em conta os intervalos de representação dos números no computador, na intenção de solucionar o problema da falta de precisão dos dados da máquina. Nessa metodologia, a solução encontrada para resolver o problema da influência da propagação do erro computacional foi trabalhar com intervalos gerados a partir do arredondamento para cima e para baixo da condição inicial x_0 da equação (1.1), de modo que os limites do primeiro intervalo fossem os pontos menor e maior

mais próximos a x_0 representáveis pela máquina. Após a definição do intervalo inicial, as operações são realizadas com cada limite, sendo também arredondadas de modo que o tamanho do intervalo nunca fique nulo.

O propósito do trabalho abordado em Rodrigues Junior et al. (2016a) foi comparar dois métodos de detecção de pontos fixos em funções recursivas que utilizam aritmética intervalar, além do *Intlab*, um *toolbox* do *Matlab* que trabalha com essa aritmética. Os métodos comparados são os descritos em Nepomuceno (2014), Rodrigues Junior e Nepomuceno (2015) e Rump (1999). As comparações são feitas através do custo computacional e tamanho dos intervalos finais de cada método, onde se verifica que apesar de todos atingirem os resultados esperados, existem diferenças nas precisões obtidas em cada método.

Finalmente, em Rodrigues Junior et al. (2016b), a análise intervalar é usada com o propósito de limitar as soluções exatas da equação (1.1) para alguns parâmetros e condições iniciais, de modo que o principal objetivo do trabalho é fazer uso da interseção de diferentes soluções obtidas por funções intervalares matematicamente equivalentes para reduzir os limites dessas soluções.

1.3 Organização do Trabalho

Este trabalho está dividido em cinco capítulos. O Capítulo 2 conta com uma breve apresentação de trabalhos que abordam o uso da análise intervalar em diversos sistemas. É importante mencionar que os trabalhos abordados nesse capítulo foram adotados como material de consulta para a elaboração desta dissertação e optou-se pela escolha de trabalhos que apresentassem números consideráveis de citações em plataformas de pesquisa de referências bibliográficas, como o *scopus*[1] e o *web of science*[2], de modo que os trabalhos escolhidos fossem relevantes para a comunidade científica. A prioridade de escolha dos artigos usados para consulta na elaboração deste trabalho não foi baseada apenas na quantidade de citações. O ano da publicação também foi um fator a ser considerado, de modo que artigos publicados na última década mostram os últimos avanços no campo de pesquisa, além de apresentarem o fato de que o assunto trabalhado na pesquisa ainda é um tema estudado pela comunidade científica.

No Capítulo 3 são apresentados de forma clara e concisa os conceitos fundamentais para

[1] http://scopus.com
[2] http://webofknowledge.com

uma fácil compreensão e maior acessibilidade ao trabalho. Inicialmente são apresentados conceitos básicos da computação numérica, com um maior detalhamento da norma IEEE 754-2008 para aritmética de ponto flutuante (IEEE, 2008). Além disso, támbém são abordadas algumas definições importantes a respeito de sistemas dinâmicos não-lineares, funções recursivas, mapa logístico, pontos fixos, assim como órbitas e pseudo-órbitas. Ao final do capítulo, os principais aspectos, propriedades e ferramentas da análise intervalar são apresentados.

O Capítulo 4 traz a metodologia empregada para a realização deste trabalho. O que se pretende neste capítulo é realizar uma fusão entre os tópicos abordados nas principais publicações em congressos realizadas ao longo do desenvolvimento desta dissertação, com o intuito de desenvolver um método intervalar eficaz para o cálculo de pontos fixos e órbitas periódicas de mapas discretos.

No Capítulo 5 são apresentados os resultados da aplicação da metodologia desenvolvida, onde são realizadas comparações desses resultados com outros métodos de análise intervalar. Ao final do capítulo, os dois métodos desenvolvidos no trabalho são comparados.

Finalmente, o Capítulo 6 apresenta as conclusões e considerações finais do trabalho desenvolvido. Além disso, também são apontadas perspectivas de trabalhos futuros.

CAPÍTULO 2

Revisão de Literatura

Durante as últimas décadas, o papel da aritmética intervalar teve um crescimento importante graças à introdução das propriedades intervalares e ferramentas que facilitam a implementação dessa aritmética no computador, como controle nos modos de arredondamentos e precisão variável, por exemplo (Alefeld e Mayer, 2000). A versatilidade da análise intervalar pode ser encontrada em diversas linhas de pesquisa, seja limitando soluções de vários problemas numéricos ou provando que tais problemas não podem ter uma solução em determinado domínio particular .

Neste capítulo, procura-se apresentar algumas referências com relevância que trabalham com a técnica da análise intervalar e lidam com erros computacionais. Os trabalhos estão divididos em seções: as referências da primeira seção são brevemente apresentadas e as demais seções são trabalhos importantes na elaboração dessa dissertação apresentados de forma mais detalhada.

2.1 Trabalhos Relevantes

O trabalho de Alefeld e Claudio (1998) é um artigo com 47 citações[1] na área de análise intervalar. Primeiramente, os autores apresentam noções de notação e exibem alguns conceitos, teoremas e exemplos a respeito das propriedades básicas da aritmética intervalar. Depois disso, o trabalho apresenta a biblioteca *XSC* (*extended scientific computation*) para algumas linguagens de programação. São detalhadas as principais características dos ambientes de programa-

[1] http://scopus.com

ção *PASCAL-XSC* (Klatte et al., 2012a), *C-XSC* (Klatte et al., 2012b), *ACRITH-XSC* (Walter, 1993a) e *FORTRAN-XSC* (Walter, 1993b).

Já Hargreaves (2002) foca em versões intervalares de alguns problemas importantes da análise numérica. São discutidas, testadas e comparadas uma variedade de técnicas de solução de sistemas intervalares de equações lineares e não-lineares, de modo que as soluções englobem as raízes desses sistemas. Além disso, a aritmética intervalar é usada para levar em conta erros de arredondamento nos cálculos computacionais da constante de Viswanath, que é dada pela taxa de crescimento da sequência aleatória de Fibonacci (Viswanath, 2000). Um ponto a se considerar é que o autor disponibiliza todos os códigos utilizados no trabalho.

O trabalho apresentado por Grigoletti et al. (2006) realiza a análise de circuitos elétricos com a aplicação da aritmética intervalar, em que as incertezas dos dados de entrada são as tolerâncias apresentadas pelos resistores desses circuitos. O problema é modelado com a utilização da técnica de análise nodal para circuitos elétricos, gerando um sistema de equações lineares intervalares, nas quais as tensões nodais do circuito são as incógnitas. A implementação do método desenvolvido no trabalho foi realizada na linguagem *Python*, produzindo uma ferramenta denominada *Free Interval Circuit Analyser* (*FINCA*). Os resultados apresentados para algumas topologias de circuitos foram comparados com os obtidos pelo *software PSpice* (Tuinenga, 1995). Em ambas as comparações, foi verificado que as tensões nodais apresentadas pelo *software* comercial se encontravam dentro da faixa de valores dos resultados intervalares produzidos pela ferramenta *FINCA*.

Soares et al. (2009) apresentam um algoritmo evolucionário intervalar para otimização robusta multiobjetivo, conhecido como *[I]RMOEA*, que é um método híbrido que combina técnicas de análise intervalar, para lidar com incertezas de uma maneira determinística, e um algoritmo evolucionário multi-objetivo. O *[I]RMOEA* é aplicado em três funções de teste e os resultados mostram que o método é uma ferramenta adequada de resolver problemas de otimização robustos com técnicas evolucionárias.

Rothwell e Cloud (2012) apresentam uma técnica intervalar de análise de erro desenvolvida em *Intlab* (Rump, 1999). Alguns exemplos são considerados e os erros numéricos obtidos são comparados com métodos de análise de erro padrão, onde se observa que em casos envolvendo fórmulas matemáticas mais complexas, a abordagem intervalar apresenta estimações de erros com muito menos esforço. Além disso, dois experimentos de laboratório são examinados e os

dados medidos são usados para explorar a aplicabilidade da análise intervalar sob condições experimentais típicas. Ao final, o método obtém resultados satisfatórios no que se propõe e os autores salientam a importância da análise de erros no ensino de engenharia e a facilidade de implementação do método intervalar desenvolvido.

O trabalho de Galias (2013) apresenta métodos computacionais para um estudo rigoroso de dois sistemas não-lineares: o mapa de Hénon (Hénon, 1976) e o circuito de Chua (Chua e Lin, 1990). Primeiramente, o artigo faz uma análise específica do mapa de Hénon para mostrar que mesmo para sistemas muito simples, a propagação dos erros de arredondamento pode levar a resultados completamente equivocados. Depois de apresentar as principais propriedades e características da análise intervalar, o trabalho apresenta métodos intervalares para o cálculo de trajetórias e representação destes sistemas dinâmicos não-lineares. O artigo mostra que muitas ferramentas de análise intervalar podem ser usadas no estudo de sistemas dinâmicos e na formulação de teoremas sobre esses sistemas. Ao final, o autor frisa que apesar dos desenvolvimentos descritos no trabalho, ainda existem muitos problemas a serem explorados na área da análise numérica de sistemas não-lineares.

O artigo de Pereira e Da Costa (2014) apresenta um método intervalar no cálculo de fluxo de potência de linhas de transmissão. A partir de dados de carga com erros associados, é realizado o cálculo do máximo ponto de carregamento (*maximum loading-point*). Além disso, também são obtidas as principais variáveis do sistema correspondentes à esse ponto: módulos de tensão, ângulos das fases, geração de potências ativa e reativa, além do fluxo de corrente e as perdas da linha. A metodologia proposta é implementada no *Intlab* (Rump, 1999) e são usados dois sistemas de potência para sua validação. Os resultados obtidos são comparados com outros do Método de Monte Carlo (Doucet et al., 2001) para os mesmos casos.

Banerjee et al. (2015) apresentam uma metodologia para alcançar o controle robusto de um Reator Nuclear PHWR (Pressurized Heavy-Water Reactor) de 540 MWe, sob variações na demanda de potência nominal e incertezas na medição da potência do reator, dos coeficientes de reatividade e dos parâmetros de transferência de calor. A metodologia usa uma abordagem intervalar e pode ser usada para sintonizar controladores PID convencionais com o intuito de realizar o controle robusto do reator nuclear. O controlador obtido por esta metodologia é um controlador ótimo e os limites aceitáveis nos parâmetros de tempo de resposta de ciclo fechado podem ser especificados diretamente. A robustez dos controladores foi estabelecida utilizando

simulação em *Matlab*.

O trabalho de Oliveira et al. (2016) faz uma abordagem intervalar para reconfiguração ótima de sistemas elétricos de distribuição para minimizar perdas de energia considerando incertezas tanto na demanda de carga quanto na geração eólica devido à intermitência da velocidade do vento. Neste trabalho, o fluxo de potência intervalar é usado para avaliar as soluções do algoritmo de otimização e minimizar a perda do sistema. O método de otimização intervalar desenvolvido neste trabalho obteve resultados similares e até melhores no cálculo das perdas do sistema em relação a outros métodos que também consideram incertezas.

2.2 Análise de Erros em Funções Recursivas

Como já mostrado anteriormente na Figura 1.1, Nepomuceno (2014) mostra que uma simples sequência de iterações no cálculo do mapa logístico (May, 1976) pode gerar resultados equivocados. Com isso, o trabalho apresenta um método intervalar cuidadoso para calcular a propagação do erro nos cálculos computacionais de funções recursivas e avalia até que ponto a resposta obtida é confiável.

O método de Nepomuceno (2014) consiste na determinação de um erro $\delta_n > 0$ relacionado a cada operação aritmética realizada pelo computador, a cada saída x_n de um mapa recursivo. Seja $\mathbb{D} \subset \mathbb{Q}$ o conjunto dos números que podem ser armazenados no computador, para o intervalo $I \subseteq \mathbb{R}$, com $x \in I$ e $f_n : I \to \mathbb{R}$ sendo uma função iterativa, existe um outro intervalo $J \subseteq D$, com $\hat{x} \in J$ sendo uma representação de x e $\hat{f}_n$ sendo uma aproximação de f_n. Com isso, tem-se que

$$d(f_n(x),\hat{f}_n(\hat{x})) \leq \delta_n. \tag{2.1}$$

A partir de (2.1), pode-se afirmar que $f_m(x) \neq f_n(x)$ somente se

$$d(\hat{f}_m(\hat{x}),\hat{f}_n(\hat{x})) > \delta_m + \delta_n. \tag{2.2}$$

O autor também apresenta uma definição sobre a convergência da resposta a pontos fixos em funções recursivas, como se segue.

Definição 2.2.1 (Nepomuceno, 2014)**.** *Se* $d(\hat{f}_n(\hat{x}^*), \hat{f}_{n-1}(\hat{x}^*)) \leq \delta_n + \delta_{n-1}$*, então* $\hat{x}^*$ *é um ponto*

fixo.

O método é aplicado no mapa logístico (Eq. (1.1)) para os mesmos parâmetros especificados na Figura 1.1, $r = 327/100$ e $x_0 = 100/327$. Através da análise dos resultados, mostrados na Tabela 2.1, pode-se concluir que já na segunda iteração não é possível garantir confiabilidade do resultado obtido, uma vez que $d(\hat{x}_2,\hat{x}_1) < \delta_2 + \delta_1$.

Tabela 2.1: Simulação das três primeiras iterações do cálculo do mapa logístico para $r = 327/100$ e $x_0 = 100/327$ com o método de Nepomuceno (2014).

n	$\widehat{x}_n$	$d(\widehat{x}_n, \widehat{x}_{n-1})$	δ_n
0	0,305810397553517	0	$2{,}77555756156289 \times 10^{-17}$
1	0,694189602446483	0,388379204892966	$3{,}25197734863617 \times 10^{-16}$
2	0,694189602446483	$2{,}22044604925031 \times 10^{-16}$	$1{,}52284898079424 \times 10^{-15}$
3	0,694189602446483	$3{,}33066907387547 \times 10^{-16}$	$5{,}43916855498739 \times 10^{-15}$

Portanto, pela análise da Tabela 2.1 e conforme a Definição 2.2.1, para os parâmetros especificados, o mapa logístico apresentou convergência para ponto fixo.

2.3 Análise de Erros com o Polinômio NARMAX

Nepomuceno e Martins (2016) apresentam uma técnica de cálculo do limite inferior de erro (*lower bound error*) na simulação do polinômio NARMAX (*Nonlinear AutoRegressive Moving Average model with eXogenous input*), baseada no conceito de extensões intervalares (Moore et al., 2009). Quando um cálculo computacional apresenta o limite inferior de erro maior do que a precisão esperada, não existe mais confiabilidade nos resultados e outras medidas devem ser tomadas. Uma delas, por exemplo, seria aumentar a precisão do *software*.

Os autores iniciam o trabalho com definições importantes sobre o polinômio NARMAX, funções recursivas, extensões intervalares e também mostram a questão de órbitas e pseudo-órbitas. A maioria dessas definições também serão vistas ao longo desta dissertação, por esse motivo, essa seção dará mais evidência ao método do limite inferior de erro desenvolvido. O limite inferior de erro $\delta_{\alpha,n}$ é definido por:

$$\delta_{\alpha,n} = \frac{|\hat{x}_{a,n} - \hat{x}_{b,n}|}{2}, \tag{2.3}$$

onde $\hat{x}_{a,n}$ e $\hat{x}_{b,n}$ são as respostas obtidas de duas funções recursivas matematicamente equivalentes, denotadas por a e b, na n-ésima iteração.

Através de teoremas, os autores demonstram que pelo menos uma das duas respostas obtidas, conhecidas como pseudo-órbitas, conceito que será detalhado no próximo capítulo, deve apresentar erro maior ou igual ao limite inferior de erro. Portanto, se o limite inferior de erro obtido em alguma iteração n é maior do que a precisão requerida, deve-se analisar melhor antes de prosseguir com a simulação. Para garantir resultados confiáveis na simulação, deve-se provar que pelo menos uma das pseudo-órbitas possui a precisão requerida.

Além disso, também é definido um critério de parada de acordo com uma tolerância mínima, definida pelo usuário. Toda vez que a precisão relativa dos cálculos $\epsilon_{\alpha,n}$ for maior do que a tolerância estipulada, a simulação deve ser interrompida. Em todos os casos analisados no artigo, a tolerância estipulada foi $\epsilon = 0{,}001$ e a precisão $\epsilon_{\alpha,n}$ é dada por:

$$\epsilon_{\alpha,n} = \frac{\delta_{\alpha,n}}{\frac{|\hat{x}_{a,n}+\hat{x}_{b,n}|}{2}} = \frac{|\hat{x}_{a,n} - \hat{x}_{b,n}|}{|\hat{x}_{a,n} + \hat{x}_{b,n}|} \tag{2.4}$$

O método foi aplicado no mapa senoidal (Nepomuceno et al., 2003), circuito de Chua (Chua e Lin, 1990) e oscilador de Duffing-Ueda (Billings, 2013). Todos as simulações foram realizadas no *Scilab 5.5.2* e os resultados foram comparados com cálculos simbólicos obtidos com o *software wxMaxima 13.04.2* com 1000 dígitos de precisão. É importante mencionar que os autores disponibilizaram todas as rotinas, além de informações do $hardware$ utilizado.

Para uma análise mais simples, serão apresentados os resultados obtidos apenas para o mapa senoidal. As duas funções polinomiais *NARMAX* matematicamente equivalentes do mapa seno são apresentadas no conjunto de equações 2.5. A Figura 2.1 apresenta os resultados para cada função nas primeiras 60 iterações e também mostra o gráfico do erro $\epsilon_{\alpha,n}$ relativo das primeiras 100 iterações em escala logarítmica.

$$\begin{aligned} G(X_n) &= 2{,}6868X_n - 0{,}2462X_n^3 \\ H(X_n) &= 2{,}6868X_n - (0{,}2462X_n)X_n^2 \end{aligned} \tag{2.5}$$

Pela análise da Figura 2.1, pode-se observar que ambas as funções $G(X_n)$ (-x-) e $H(X_n)$ (-o-) iniciam com a mesma condição inicial e o erro ultrapassa a tolerância estipulada com 44 iterações.

Os outros dois casos abordados, apresentam resultados análogos, porém o número de iterações onde a precisão relativa dos cálculos ultrapassou a tolerância estipulada foi diferente para cada mapa: o circuito de Chua alcançou 464 iterações e o oscilador de Duffing-Ueda 3345. Isso

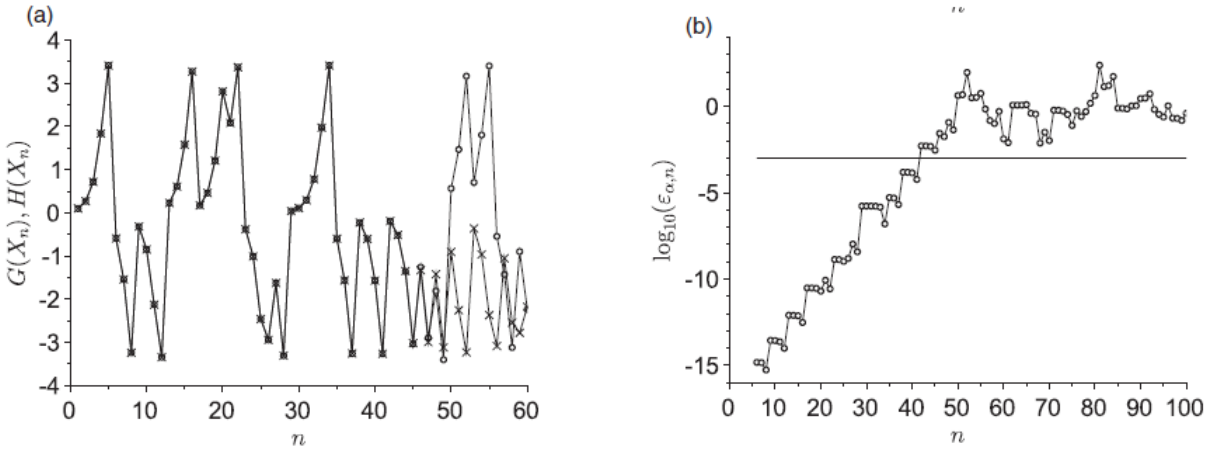


Figura 2.1: Mapa senoidal: (a) 60 primeiras iterações dos cálculos de $G(X_n)$ (-x-) e $H(X_n)$ (-o-) e (b) evolução do erro relativo comparada com a tolerância estipulada.

indica que a simulação deveria ser interrompida e melhor analisada nesses pontos.

Além disso, para todos os casos o limite inferior de erro foi calculado na décima iteração e comparado com o valor obtido por cálculos simbólicos, onde foi observado que ambos os erros associados à resposta de cada função são maiores ou iguais ao limite inferior de erro, concordando com o que é proposto no trabalho.

2.4 Conclusões do Capítulo

A computação intervalar tem importância fundamental na análise de erros computacionais ou incertezas acerca de dados de diversos sistemas. Ela apresenta uma faixa de valores onde se encontra a resposta exata de um determinado cálculo numérico, garantindo que essas incertezas ou erros numéricos não anulem os resultados obtidos.

Sistemas dinâmicos não-lineares ainda não tem recebido a atenção que deveriam em relação à propagação de erros devido a arredondamentos e truncamentos, que são inerentes à grande maioria dos computadores atuais. Portanto, evidencia-se o fato de que trabalhos acadêmicos com cálculos numéricos devem disponibilizar todo o processo e informações usadas para se chegar ao resultado apresentado. Yabuki e Tsuchiya (2013) salientam o fato de que raramente são incluídos o tipo de computador, a linguagem de programação, os códigos e muito menos o modo computacional de precisão utilizados. Além disso, os autores acreditam que um dos objetivos principais de artigos científicos deve ser a reprodutibilidade dos resultados neles reportados.

Por fim, apesar da importância e versatilidade da análise intervalar, nas plataformas de pesquisa de referências bibliográficas não foram encontrados outros métodos intervalares para simulação rigorosa do mapa logístico, fato que motivou o desenvolvimento deste trabalho.

CAPÍTULO 3

Fundamentação Teórica

3.1 Computação Numérica

Grande parte dos problemas numéricos são calculados por meio de ferramentas computacionais. Contudo, um fato que é pouco explorado no meio científico é que os computadores não são capazes de representar exatamente todos os infinitos números reais. Essa seção apresenta alguns pontos fundamentais da norma IEEE 754-2008 de ponto flutuante (IEEE, 2008), responsável por assegurar um padrão na representação e nas operações aritméticas dos números reais nos computadores.

3.1.1 Norma IEEE 754-2008 de Ponto Flutuante

O ponto flutuante é usado em computadores desde meados da década de 1950, onde durante as duas décadas seguintes cada empresa adotava um padrão diferente. Isso acarretava em incompatibilidade na forma como um programa se comportava em máquinas diferentes. Buscando uma solução ao problema, nas décadas de 1970 e 1980, um grupo de cientistas e engenheiros, liderados pelo *Institute for Electrical Engineering and Electronics Engineers – IEEE*, desenvolveu um padrão para a representação e operação aritmética de ponto flutuante (Overton, 2001).

Portanto, o padrão IEEE 754-2008 contém normas a serem seguidas pelos fabricantes de computadores e consultores de softwares no tratamento da aritmética computacional para números de ponto flutuante. Assim, essa norma é responsável por conceitos que serão mais detalhados à frente, como modos de arredondamento, armazenamento dos números no computador,

ocorrência de *underflow* e *overflow*, além da realização de operações aritméticas básicas computacionais (Viana, 1999).

3.1.2 Noções da Representação dos Números no Computador

Um número real é armazenado no computador como ponto flutuante, que é um formato de representação digital de números reais usado nos computadores. O termo ponto flutuante deve-se ao fato de que o ponto ou a vírgula decimal do número se move para a posição posterior ao primeiro dígito não nulo (Campos e Lima, 2005). Um número x é representado como ponto flutuante da seguinte forma:

$$x = \pm S \times B^E, \tag{3.1}$$

em que o número S é chamado de significante ou mantissa, E é o expoente e B a base utilizada.

A maioria dos computadores atuais utilizam base binária (B=2), porém outras bases já foram utilizadas no passado, tais como a hexadecimal (B=16) em computadores IBM e trinária (B=3) em computadores russos (Brousentsov, 1960). Por razões óbvias, para facilitar a interação humana, um computador geralmente converte valores numéricos a partir de notação decimal na entrada, além de converter para notação decimal na saída de dados, independentemente da base que utiliza internamente (Heath, 2002).

Nos computadores atuais que seguem a norma IEEE 754-2008, se utiliza internamente a base 2, com algarismos 0 e 1 (Weber, 2000). A expansão binária do significante de um número x diferente de zero é dada por

$$S = (b_0,b_1b_2...) \quad com \quad b_0 = 1. \tag{3.2}$$

Como visto na equação 3.2, a norma IEEE 754-2008 define que o primeiro bit do significante é sempre diferente de zero para números que podem ser representados como ponto flutuante. Geralmente, esse primeiro bit é chamado de bit escondido pelo fato de o computador não usar sua memória para armazená-lo. Isso assegura a unicidade de representação e mantém a precisão do computador a maior suportada possível. O número zero não pode ser normalizado e tem representação especial, com mantissa nula e expoente menor possível (Viana, 1999). Os bits após o ponto binário são chamados de parte fracionária do significante e as equações (3.1) e (3.2) são chamadas representações normalizadas de um número real (Overton, 2001).

3.1.3 Representação Computacional em Diferentes Formatos da Norma IEEE 754-2008

A norma IEEE 754-2008 especifica dois formatos básicos para o computador: o formato *single*, que usa 32 bits e o *double* que usa 64 bits. O formato *single* disponibiliza 1 bit para o armazenamento do sinal, 8 bits para o expoente e 23 bits para o significante. Já o formato *double* apresenta 1 bit para armazenar o sinal, 11 bits para o expoente e 52 bits para o significante. O sinal sempre é representado com 1 bit e pode ser negativo se o bit tem valor 1 ou positivo se o bit tem valor 0 (Overton, 2001). A tabela a seguir mostra a disposição dos termos para essas configurações:

Tabela 3.1: Disposição dos bits segundo padrão IEEE 754-2008 para sistemas de 32 e 64 bits.

Sistema	Sinal ($\pm$)	Expoente (E)	Mantissa (S)
Single (32 bits)	1 bit	8 bits	23 bits
Double (64 bits)	1 bit	11 bits	52 bits

Por apresentar uma maior quantidade de bits para armazenar dados, o sistema *double* apresenta mais precisão e uma faixa maior de números representáveis em relação do *single*. A faixa de valores que o sistema de 64 bits pode representar vai de aproximadamente $2{,}2 \times 10^{-308}$ até $1{,}8 \times 10^{308}$, enquanto o sistema de 32 bits apresenta uma faixa de valores que vai aproximadamente de $1{,}2 \times 10^{-38}$ até $3{,}4 \times 10^{38}$.

Números maiores que zero e menores do que o menor valor possível de ser representado pelo formato são considerados iguais a zero pelo computador. A esse arredondamento, dá-se o nome de *underflow*. Da mesma forma, números que ultrapassam o limite superior do formato da máquina, são considerados infinito e essa ocorrência é chamada de *overflow*. Essa faixa de valores é simétrica, ou seja, vale tanto para números positivos quanto para negativos (IEEE, 2008).

3.1.4 Precisão do Sistema

A incapacidade de representação de dados apresentada na subseção anterior não existe apenas para números que ultrapassam os limites inferiores ou superiores da máquina. A quantidade de números reais que podem ser representados de forma precisa dentro da faixa limitada do formato adotado depende da precisão do sistema. A precisão de um sistema de ponto flutuante depende da quantidade de bits representáveis pelo formato, sendo dada pelo número de bits do

significante, incluindo o bit escondido (Overton, 2001). Um ponto flutuante com precisão p é expresso por:

$$x = \pm(1{,}b_1 b_2 ... b_{p-2} b_{p-1})_2 \times 2^E \tag{3.3}$$

O menor ponto flutuante x maior do que 1 é dado por:

$$x = (1{,}000...1)_2 = 1 + 2^{-(p-1)} \tag{3.4}$$

A distância entre esse número e 1 é chamada de épsilon da máquina (ϵ).

$$\epsilon = (0{,}000...01)_2 = 2^{-(p-1)} \tag{3.5}$$

3.1.5 ULP (*Unit in the Last Place*) da Máquina

Para qualquer ponto flutuante dado, pode-se determinar a distância dele para o próximo número que a máquina é capaz de armazenar. Essa distância é chamada de ulp (*unit in the last place*) e é dada por (IEEE, 2015):

$$ulp(x) = \epsilon \times 2^E \tag{3.6}$$

Nenhum dos números existentes entre os pontos flutuantes x e $y = x + \frac{1}{2}ulp(x)$ podem ser armazenados pelo computador, então seus valores são arredondados para um desses dois números. Portanto, se o resultado z de um cálculo é o ponto flutuante mais próximo ao resultado correto, ele pode apresentar um erro de até $\frac{1}{2}ulp(z)$ (Goldberg, 1991).

Analisando as equações (3.3) e (3.6), pode-se observar que quanto maior o valor de x, maior também é sua ulp, ou seja, maior é a faixa de valores ao redor de x não representáveis pelo computador.

Exemplo 3.1

A ulp do número 4 na base 10, ou $(4)_{10}$, representado na base binária por $(100)_2$, ou ainda $(1{,}00 \times 2^2)_2$, em um sistema de 64 bits, é:

$$B = 2, \qquad p = 53, \qquad E = 2,$$

$$ulp(4_{10}) = ulp(100_2) = 2^{-(p-1)} \times 2^E = 2^{-(53-1)} \times 2^2 = 8{,}8818 \times 10^{-16}$$

Exemplo 3.2

Para um sistema com formato *double* (64 bits), $1 \times 10^{50} + 1 \times 10^{30} = 1 \times 10^{50}$. O cálculo da ulp de 1×10^{50} explica o fato do resultado dessa soma não ser preciso.

$$ulp(1 \times 10^{50}) \approx 2 \times 10^{34} \tag{3.7}$$

Como $\frac{1}{2}ulp(1 \times 10^{50}) \approx 1 \times 10^{34} > 1 \times 10^{30}$, tem-se que a faixa de valores não representáveis ao redor de 1×10^{50} engloba o resultado da soma, arredondando o resultado para 1×10^{50}.

3.1.6 Arredondamento

Além do zero, existem outros números que não podem ser normalizados, pois a normalização iria resultar em um expoente fora da faixa de representação suportada pelo formato da máquina. Esses números usam a combinação de expoente zero e a parte fracionária não zero e são chamados de números subnormalizados. Números subnormalizados apresentam precisão reduzida à medida que decrescem (IEEE, 2008).

Em suma, um número real na norma IEEE 754-2008 de ponto flutuante pode ser expresso por $x = \pm(b_0{,}b_1b_2...b_{p-2}b_{p-1})_2 \times 2^E$, em que p é a precisão do sistema de ponto flutuante, $b_0 = 1$ e $E_{min} \leq E \leq E_{max}$ para um número normalizado ou $b_0 = 0$ e $E = E_{min}$ para um número subnormalizado. Para o formato *single*, a norma estabelece que $p = 24$, $E_{min} = -126$ e $E_{max} = 127$, enquanto que para o formato *double*, tem-se $p = 53$, $E_{min} = -1022$ e $E_{max} = 1023$ (Faranda et al., 2012).

Denotando-se N_{min} e N_{max} como menor e maior valores positivos normalizados que a máquina pode representar, tem-se que um número real não é ponto flutuante se alguma das duas afirmativas é verdadeira (Overton, 2001):

i) x está fora da faixa normalizada, ou seja, é maior do que N_{max}, menor do que $-N_{max}$, está entre 0 e N_{min} ou ainda entre $-N_{min}$ e 0. Por exemplo, $x = 2^{130}$ ou $x = 2^{-130}$ estão fora da faixa normalizada em um sistema que adota o formato *single* de 32 bits;

ii) A expansão binária do número requer mais do que p bits para especificar exatamente o número, ou seja, a precisão de ponto flutuante p não é suficiente para representar o número

x exatamente. Por exemplo o número $x = 1+2^{-25} = (1{,}00000000000000000000000001)_2$ requer mais bits que o formato *single* permite.

Em ambos os casos, é necessário aproximar x por uma representação diferente do número real original, arredondando esse valor para um ponto flutuante. Segundo IEEE (2008), o arredondamento do número x pode ser realizado seguindo uma das regras abaixo:

- Arredondamento Para o Mais Próximo: o número real x é aproximado para o ponto flutuante que é mais próximo a ele.
- Arredondamento Negativo ou Para Baixo: o número real x é aproximado para o ponto flutuante que não é maior e que é mais próximo a ele.
- Arredondamento Positivo ou Para Cima: o número real x é aproximado para o ponto flutuante que não é menor e que é mais próximo a ele.
- Arredondamento em Direção ao Zero: funciona da mesma forma que o arredondamento negativo se $x > 0$ e o arredondamento positivo se $x < 0$.

É importante mencionar que o arredondamento para o mais próximo é geralmente adotado como o modo padrão de arredondamento e, caso os pontos flutuantes mais próximos estejam distanciados igualmente do número real, aquele com o último bit significativo igual a zero é escolhido. Além disso, a notação utilizada para demonstrar o arredondamento de x será $round(x)$ e se x é ponto flutuante, então $round(x) = x$.

3.1.7 Erros Computacionais

A análise de erros e a preocupação com precisão numérica vem sendo tema de interesse desde antes da invenção do computador (Richardson, 1911). No entanto, depois que o computador passou a ser utilizado, houveram avanços significativos em métodos numéricos que envolvem essa área (Freitas, 2002). Uma das características fundamentais da computação numérica é que, embora muitos programas e cálculos apresentem resultados com precisão, pode acontecer que erros computacionais influenciem significativamente no resultado (Hickey et al., 2001).

A distinção entre os principais tipos de erros computacionais é importante para o entendimento do comportamento de algoritmos numéricos e dos fatores que afetam a precisão de

resultados. Como já mencionado, os erros computacionais podem ser divididos em erros de truncamento e arredondamento Heath (2002):

- Erro de Truncamento: é o erro devido ao truncamento de séries infinitas, substituição de equações diferenciais por equações de diferenças, representação de uma função arbitrária por um polinômio ou parada de uma sequência iterativa antes da convergência.

- Erro de Arredondamento: é a diferença entre o resultado produzido por um cálculo usando aritmética dos números reais e o resultado produzido pelo mesmo cálculo usando aritmética de ponto flutuante. Como falado na seção anterior, os computadores apresentam precisão limitada na representação dos números reais e realização de operações aritméticas.

3.1.8 Comentários Finais da Seção

A aritmética de ponto flutuante é amplamente utilizada na computação científica. Enquanto muitos programadores estão cientes que o conjunto dos números de ponto flutuante são uma aproximação do conjunto dos números reais, apenas alguns são conhecedores dos perigos que isso pode acarretar para a programação (Leeser et al., 2014).

Grande parte das operações matemáticas que envolvem números reais necessitam de infinitos dígitos para serem exatamente representadas. Com a precisão finita apresentada pelo computador, cada número é representado com um número finito de dígitos. Isso faz com que o conjunto dos números reais seja significantemente distinto do conjunto dos números produzidos por medições e cálculos computacionais (Chaitin-Chatelin, 1996)

Cada operação aritmética entre dois números feita no computador pode carregar um erro. Quando o processo se repete por várias vezes, o erro de arredondamento de cada operação tende a se propagar de acordo com as iterações. Com isso, não é raro que os resultados finais estejam distantes do que se esperaria obter, ainda que todas as fases de resolução do algoritmo tenham sido realizadas corretamente (Ruggiero e Lopes, 1997).

Como já foi dito, o resultado de uma operação aritmética no computador entre dois números que podem ser representados como ponto flutuante, é dado pelo resultado exato dessa operação aritmética com um erro δ associado. O valor de δ pode ser nulo, no caso do resultado dessa operação ser um ponto flutuante, ou ainda seu módulo pode ser de até $0{,}5$ vezes a ulp do resultado exato. O modo de arredondamento define se o resultado computacional é aproximado

para um ponto flutuante maior ou menor em relação ao resultado exato, ou seja, se δ é maior ou menor que zero (Overton, 2001).

A utilização dos modos de arredondamento para obter resultados computacionais aproximados a pontos flutuantes maiores e menores ao resultado exato, faz com que uma faixa de valores a qual a resposta exata pertença, seja delimitada. Esse princípio dá origem à técnica da computação por intervalos, que virá a ser utilizada no desenvolvimento deste trabalho e melhor detalhada na próxima seção.

3.2 Análise Intervalar

A técnica de computação por intervalos foi proposta como uma ferramenta para limitar os erros de arredondamentos na computação numérica e é um campo que engloba a matemática rigorosa com a computação científica. Essa ligação entre computação e matemática intervalar, torna possível resolver problemas que não podem ser resolvidos de forma eficiente usando a aritmética de ponto flutuante (Moore et al., 2009).

Ao invés de usar um número representado por ponto flutuante como uma aproximação do valor real de uma variável no sistema sob investigação, a aritmética intervalar reconhece a precisão limitada do computador, associando a essa variável um conjunto ou intervalo de valores reais possíveis. O objetivo da computação intervalar é, portanto, investigar e garantir que todos os possíveis valores de uma resposta, devido à incertezas e erros inerentes ao sistema, estejam dentro do intervalo (Hickey et al., 2001).

Como já visto no Capítulo 2, algoritmos que usam a técnica intervalar podem ser aplicados em grande parte das áreas que englobam a análise numérica. Porém, embora não seja difícil implementá-los, os resultados podem não ter utilidade se caso os intervalos obtidos como resposta sejam grandes. Se esse é o caso, então outros algoritmos devem ser considerados, ou então devem ser desenvolvidos novos para que produzam intervalos mais estreitos (Hargreaves, 2002).

3.2.1 Breve História da Análise Intervalar

Precisar exatamente quando ocorreu o surgimento da análise intervalar não é uma tarefa fácil. Porém, alguns estudos apontam que o primeiro registro que consta a análise de intervalos

foi elaborado por Arquimedes, publicado em Heath (1897), onde o objetivo era fazer um cálculo com precisão do número π. Depois, na década de 1950, muitos trabalhos foram desenvolvidos com a ideia de limitação de erros usando análise intervalar.

No entanto, embora todos esses estudos apresentem relevância, o livro de Moore (1966) foi o primeiro trabalho com consistência, reconhecido por vários pesquisadores como uma contribuição real para a área da análise intervalar (Hargreaves, 2002).

Durante quase duas décadas após a publicação de Moore (1966), a análise intervalar não foi um tema muito investigado pela comunidade científica, talvez pelo fato de ainda não existir necessidade para sua utilização. Com o advento dos computadores, a área passou a ser mais procurada e nos anos 90 a análise intervalar tornou-se popular com a adesão de vários grupos de pesquisa em várias partes do mundo. Hoje em dia, o próprio IEEE reconhece e padroniza o uso da computação intervalar (IEEE, 2015).

3.2.2 Definições Básicas

Por um intervalo, entende-se um conjunto fechado e limitado de números reais. Pode-se considerar também o intervalo como um par ordenado de pontos limites (Moore, 1966).

Neste trabalho, tanto intervalos quanto pontos limites serão representados por letras maiúsculas, seguindo o que foi adotado em Moore et al. (2009). Os limites inferior e superior de um intervalo X serão denotados por $\underline{X}$ e $\overline{X}$, respectivamente. Portanto,

$$X = [\underline{X},\overline{X}] \tag{3.8}$$

Diz-se que o intervalo X é degenerado se $\underline{X} = \overline{X}$. Assim, um número real pode ser representado por um intervalo degenerado como $x = [x,x]$ (Alefeld e Herzberger, 2012).

Segundo Moore et al. (2009), a informação de que $y \in [59{,}62]$ é muito mais clara em relação à afirmação de que $y \approx 60$, uma vez que a resposta intervalar demonstra a ideia do tamanho do erro associado. Se é sabido que uma resposta x está contida dentro da faixa de valores correspondentes ao intervalo $[\underline{X},\overline{X}]$, pode-se usar o ponto médio $m = (\underline{X} + \overline{X})/2$ desse intervalo como uma aproximação a solução x, com $|x - m| \leq \omega/2$, onde $\omega = \overline{X} - \underline{X}$ é o tamanho do intervalo. Com isso, se obtém uma solução aproximada e um erro associado à essa aproximação. A Figura 3.1 apresenta as relações que foram definidas nessa subseção.

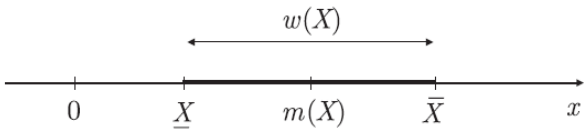


Figura 3.1: Tamanho, ponto médio e limites de um intervalo X. Adaptado de Moore et al. (2009).

Outra definição básica é que se um número x pertence a um intervalo X, significa que seu valor está entre os limites desse intervalo. Ou seja,

$$x \in X \text{ implica que } \underline{X} \leq x \leq \overline{X}. \tag{3.9}$$

É importante salientar que a qualidade de uma solução intervalar é inversamente proporcional ao tamanho do intervalo, ou seja, quanto menor o erro associado, mais precisa é a solução.

3.2.3 Relações Algébricas de Intervalos

Todos os conceitos que serão vistos nessa subseção foram retirados de Moore et al. (2009) e Rothwell e Cloud (2012). Dois intervalos X e Y são ditos serem iguais se são os mesmos conjuntos, ou ainda, isso acontece se seus pontos limites são os mesmos:

$$X = Y \quad se \quad \underline{X} = \underline{Y} \quad e \quad \overline{X} = \overline{Y}. \tag{3.10}$$

A interseção entre dois intervalos X e Y é o conjunto de números reais que pertence a ambos os intervalos. Essa interseção é vazia tanto se $\overline{Y} < \underline{X}$ ou $\overline{X} < \underline{Y}$. Nesse caso, denota-se:

$$X \cap Y = \emptyset, \tag{3.11}$$

indicando que X e Y não possuem pontos em comum. Se esse não for o caso, define-se a interseção $X \cap Y$ como o intervalo

$$X \cap Y = [max\{\underline{X},\underline{Y}\},min\{\overline{X},\overline{Y}\}]. \tag{3.12}$$

A união entre dois intervalos $X \cup Y$ é o conjunto de números reais que pertence a X ou Y (ou ambos). Se $X \cap Y$ não é vazio, então $X \cup Y$ é um intervalo dado por:

$$X \cup Y = [min\{\underline{X},\underline{Y}\},max\{\overline{X},\overline{Y}\}]. \tag{3.13}$$

Importância da Interseção

A interseção tem um papel fundamental na análise intervalar. Essa afirmativa vem do fato de que se existem dois intervalos distintos que contém um resultado de interesse, independente de como foram obtidos, a interseção, que deve ser mais estreita que ambos, também irá conter o resultado.

A afirmativa anterior pode ser simplificada através da equação (3.14) a seguir, que mostra que o tamanho da interseção entre dois intervalos tem no máximo o tamanho do menor dos intervalos.

$$\omega(X \cap Y) \leq min\{\omega(X),\omega(Y)\}. \tag{3.14}$$

Com isso, essa simples operação algébrica pode diminuir o tamanho de um intervalo no qual a solução de um problema está contida, diminuindo o erro associado à resposta.

3.2.4 Operações Aritméticas Elementares

O que é importante ressaltar nesta subseção é que realizar operações com intervalos é o mesmo do que realizar operações com conjuntos. Quando operações são realizadas entre dois intervalos, acarreta em um intervalo resultante contendo todos os resultados das operações entre pares de valores de cada intervalos. A seguir serão definidas as operações aritméticas intervalares para os intervalos X e Y (Alefeld e Herzberger, 2012; Moore et al., 2009).

Adição

$$X + Y = [\underline{X} + \underline{Y},\overline{X} + \overline{Y}]. \tag{3.15}$$

Por exemplo, seja $X = [0,2]$ e $Y = [-1,1]$, então $X + Y = [0 + (-1),2 + 1] = [-1,3]$. É importante ressaltar que essa operação não é o mesmo que $X \cup Y = [-1,2]$.

Subtração

$$X - Y = [\underline{X} - \overline{Y},\overline{X} - \underline{Y}]. \tag{3.16}$$

Se $X = [-1,0]$ e $Y = [1,2]$, então $X - Y = [-1 - 2,0 - 1] = [-3, -1]$. Ressalta-se que a operação $X - Y$ é o mesmo que $X + (-Y)$.

Multiplicação

$$X.Y = [\min\{S\}, \max\{S\}], \tag{3.17}$$

onde $S = \{\underline{X}.\underline{Y},\underline{X}.\overline{Y},\overline{X}.\underline{Y},\overline{X}.\overline{Y}\}$.

Sejam, por exemplo, $X = [-1,0]$ e $Y = [1,2]$. Então $S = \{-1.1, -1.2,0.1,0.2\} = \{-1, -2,0\}$. Portanto, $X.Y = [\min\{S\}, \max\{S\}] = [-2,0]$.

Da mesma forma que a multiplicação entre dois intervalos, a multiplicação de um intervalo por um escalar pode ser feita. Basta considerar o escalar como um intervalo degenerado. No caso analisado, $2.Y = [2,2].[1,2] = [2,4]$.

Divisão

$$\frac{X}{Y} = X.\left(\frac{1}{Y}\right), \tag{3.18}$$

onde $1/Y = [1/\overline{Y},1/\underline{Y}]$. Nesse caso, deve-se assumir que $0 \notin Y$.

Por exemplo, se $X = [4,6]$ e $Y = [1,2]$, então $X/Y = [4,6].[1/2,1]$. Nesse caso, $S = \{2,4,3,6\}$. Como $X/Y = X.(1/Y) = [\min\{S\}, \max\{S\}] = [2,6]$.

Definição Básica de um Intervalo

Qualquer intervalo X pode ser definido por

$$X = m(X) + \left[-\frac{1}{2}\omega(X),\frac{1}{2}\omega(X)\right] \tag{3.19}$$

3.2.5 Propriedades Algébricas Intervalares

As definições das propriedades algébricas vistas nesta seção foram retiradas de Moore et al. (2009).

Comutatividade e Associatividade

Na aritmética intervalar, tanto a adição quanto a multiplicação apresentam propriedades comutativa e associativa. O conjunto de equações (3.20) a seguir mostra essas propriedades para três intervalos X, Y e Z.

$$\begin{aligned} X + Y &= Y + X, \\ X + (Y + Z) &= (X + Y) + Z, \\ X.Y &= Y.X, \\ X.(Y.Z) &= (X.Y).Z. \end{aligned} \tag{3.20}$$

Elementos Neutros da Adição e Multiplicação

Para essa aritmética, os intervalos degenerados 0 e 1 são os elementos neutros da adição e multiplicação, respectivamente. O conjunto de equações (3.21) mostra essas propriedades para um intervalo X.

$$\begin{aligned} 0 + X &= X + 0 = X, \\ 1.X &= X.1 = X, \\ 0.X &= X.0 = 0. \end{aligned} \tag{3.21}$$

Elemento Inverso

Não existem elementos inversos aditivos nem multiplicativos na aritmética intervalar. Com isso, $-X$ não é um elemento inverso aditivo de um intervalo X. De fato, tem-se que

$$X + (-X) = [\underline{X},\overline{X}] + [-\overline{X}, -\underline{X}] = [\underline{X} - \overline{X},\overline{X} - \underline{X}], \tag{3.22}$$

e essa soma é igual a $[0,0]$ somente se $\underline{X} = \overline{X}$.

De maneira similar, $X/X = 1$ somente se $\omega(X) = 0$. De maneira geral,

$$X/X = [\underline{X}/\overline{X},\overline{X}/\underline{X}] \quad se \quad 0 < \underline{X}, \quad e \tag{3.23}$$

$$X/X = [\overline{X}/\underline{X},\underline{X}/\overline{X}] \quad se \quad \overline{X} < 0. \tag{3.24}$$

Portanto, os inversos multiplicativos e aditivos não são cumpridos na aritmética intervalar, a não ser para intervalos degenerados. No entanto, sempre os elementos neutros terão a relação: $0 \in X - X$ e $1 \in X/X$.

Subdistributividade

A propriedade distributiva da aritmética real também não é cumprida para aritmética intervalar. Um contra-exemplo simples pode ser visto utilizando os intervalos $X = [1,2]$, $Y = [1,1]$ e $Z = [-1, -1]$.

$$\begin{aligned} X.(Y+Z) &= [1,2].([1,1] - [1,1]) \\ &= [1,2].[0,0] \\ &= [0,0]. \end{aligned} \tag{3.25}$$

Ao passo que,

$$\begin{aligned} X.Y + X.Z &= [1,2].[1,1] - [1,2].[1,1] \\ = [\min\{1,2\}, \max\{1,2\}] &- [\min\{1,2\}, \max\{1,2\}] \\ &= [1,2] - [1,2] \\ &= [-1,1]. \end{aligned} \tag{3.26}$$

Entretanto, a propriedade da subdistributividade se aplica a aritmética intervalar, e possui a seguinte propriedade:

$$X.(Y+Z) \subset X.Y + X.Z. \tag{3.27}$$

A propriedade distributiva pode ser aplicada somente para $x.(Y+Z) = x.Y + x.Z$, onde Y e Z são intervalos e x é um número real. Além disso, $X.(Y+Z) = X.Y + X.Z$ se $Y.Z > 0$.

Cancelamento

A propriedade do cancelamento, dada pela equação (3.28) é assegurada para aritmética intervalar.

$$X + Z = Y + Z \Longrightarrow X = Y \tag{3.28}$$

3.2.6 Funções Intervalares

Seja f uma função de uma variável real x. A função intervalar $f(X)$ apresenta uma faixa de valores tomados por $f(x)$ de acordo com a variação de x em um intervalo X. Portanto, a imagem de uma função intervalar $f(X)$ dada pelo o conjunto representado por X mapeado por f é:

$$f(X) = \{f(x) : x \in X\}. \tag{3.29}$$

O conjunto descrito pela equação (3.29) pode ser escrito de uma maneira um pouco diferente. Por exemplo, para a função $f(x) = x^2$, com $x \in \mathbb{R}$, se $X = [\underline{X},\overline{X}]$, é evidente que o conjunto

$$f(X) = \{x^2 : x \in X\} \tag{3.30}$$

pode ser expresso por

$$f(X) = \begin{cases} (\underline{X^2}, \overline{X^2}), & \text{se} \quad 0 \leq \underline{X} \leq \overline{X} \\ (\overline{X^2}, \underline{X^2}), & \text{se} \quad \underline{X} \leq \overline{X} \leq 0 \\ (0, \max[\underline{X^2}, \overline{X^2}]), & \text{se} \quad \underline{X} \leq 0 \leq \overline{X} \end{cases}$$

Pode-se observar que a definição anterior pode apresentar um resultado distinto de $X.X$. Por exemplo, para $X = [-1,1]$, utilizando o terceiro caso da definição proposta, o intervalo resultante será $[0, \max\{-1^2,1^2\}] = [0,1]$. Enquanto que $X.X = [-1,1].[-1,1] = [-1,1]$. No entanto, $[-1,1]$ contém o intervalo $[0,1]$.

3.2.7 Superestimação de Intervalos

A superestimação dos limites no cálculo de X^2 como $X.X$ na seção anterior acontece devido ao fenômeno da dependência intervalar. A dependência intervalar é um fator crucial em cálculos computacionais que utilizam aritmética intervalar. Essa é a razão principal do fato de que simples mudanças de cálculos com ponto flutuante por intervalos podem não levar a resultados muito satisfatórios (Moore et al., 2009).

A superestimação de intervalos ocorre devido ao fato de que a aritmética intervalar considera cada intervalo independente. Mesmo quando dois intervalos iguais são subtraídos, por exemplo,

o resultado não é $[0,0]$ devido ao fato de que a operação subtração não considera que esses dois intervalos representem a mesma quantidade (Alefeld e Herzberger, 2012).

O problema da superestimação se propaga quando resultados intermediários são usados em cálculos computacionais seguintes. Galias (2013) também afirma que se uma certa variável intervalar é usada muitas vezes no mesmo cálculo, o resultado tende a ser superestimado. Esse fato pode ser facilmente mostrado para a função $f(X) = 2.X - X$, para $X = [0,1]$. Tem-se que $f([0,1]) = [0,2] - [0,1] = [-1,2]$, porém o resultado verdadeiro é $[0,1]$. Uma maneira de evitar isso, é simplificar a função de modo que a variável intervalar apareça a menor quantidade possível de vezes. Nesse caso, deveria ser feito o cálculo para $f(X) = X$.

3.2.8 Extensões Intervalares

Para uma função f definida para uma variável real x, Moore e Bierbaum (1979) apresentam a seguinte definição.

Definição 3.2.1. *Uma extensão intervalar de f é uma função intervalar F definida para mapear uma variável intervalar X, tal que para argumentos reais tem-se*

$$F(x) = f(x), \tag{3.31}$$

Exemplo 3.3

Algumas extensões intervalares de $f(x) = rx(1-x)$ podem ser dadas por:

$$\begin{aligned} F(X) &= r(X(1-X)) \\ G(X) &= rX - rX^2 \\ H(X) &= r(X - X^2) \end{aligned} \tag{3.32}$$

É importante ressaltar que as equações do conjunto (3.32) são matematicamente equivalentes, porém apresentam sequências diferentes nas operações básicas (Rudolph-Lilith e Muller, 2014). Além disso, como mostrado nas seções anteriores, as equações podem apresentar diferentes resultados devido à não aplicação da distributividade, além de inverso aditivo e multiplicativo (Moore et al., 2009).

Definição 3.2.2 (Nepomuceno e Martins, 2016)**.** *Duas extensões intervalares G e H são equivalentes se $G(X) = H(X)$ para todos os componentes de um mesmo intervalo.*

Exemplo 3.4

Sejam F, G e H três extensões intervalares dadas por:

$$\begin{aligned} F(X) &= rX - rX^2, \\ G(X) &= rX(1-X), \\ H(X) &= r(X(1-X)). \end{aligned} \tag{3.33}$$

Considerando $r = 3$ e $X = [0{,}3\ ,\ 0{,}4]$, tem-se:

$$\begin{aligned} F(X) &= 3.[0{,}3{,}0{,}4] - 3.([0{,}3{,}0{,}4]^2) = [0{,}42{,}0{,}93], \\ G(X) &= 3.[0{,}3{,}0{,}4].([1{,}1] - [0{,}3{,}0{,}4]) = [0{,}54{,}0{,}84], \\ H(X) &= 3.([0{,}3{,}0{,}4].([1{,}1] - [0{,}3{,}0{,}4])) = [0{,}54{,}0{,}84]. \end{aligned}$$

A análise deste exemplo mostra que a extensão F não é equivalente à nenhuma das outras. No entanto, G e H são extensões intervalares equivalentes.

3.2.9 Funções Racionais Intervalares

Para uma boa compreensão da metodologia desenvolvida nessa dissertação, é importante expor o conceito de funções racionais intervalares.

Definição 3.2.3 (Moore et al., 2009)**.** *Uma função racional intervalar é uma função intervalar cujo resultado pode ser obtido por uma sequência finita específica de operações aritméticas.*

Exemplo 3.5

Considere a função $F(X) = [2{,}2].X.([1{,}1] - X)$. Os cálculos de $F(X)$ podem ser divididos em sequências finitas de operações aritméticas intervalares descritas por:

$$\begin{aligned} T_1 &= [1{,}1] - X, && \text{uma subtração intervalar} \\ T_2 &= [2{,}2].X, && \text{uma multiplicação intervalar} \\ F(X_1) &= T_1.T_2, && \text{uma multiplicação intervalar.} \end{aligned}$$

Portanto, F é uma função racional intervalar. Segundo Moore et al. (2009), uma função racional intervalar geralmente deve surgir como uma extensão intervalar de alguma função real. No exemplo acima, $F(X)$ é uma extensão da função do mapa logístico, representada na forma $f(x) = rx(1 - x)$.

Exemplo 3.6

A função $F(X) = m(X) + \frac{1}{2}.(X - m(X))$ não é uma função racional intervalar. Isso ocorre porque, mesmo com seus cálculos podendo ser representados por um número finito de operações, $m(X) = (\underline{X} + \overline{X})/2$ não é uma operação aritmética intervalar.

Definição 3.2.4 (Moore et al., 2009)**.** *Se F é uma função racional intervalar e uma extensão intervalar de f, então*

$$f(X) \subseteq F(X)$$

A Definição 3.2.4 afirma que um valor obtido por F contém a faixa de valores da função real correspondente f quando a variável de f estiver dentro do intervalo em questão.

Isso quer dizer que os valores obtidos por $f(x)$ estarão contidos no intervalo $F(X)$ para todo $x \in X$.

Portanto, para um número x que está dentro da faixa do intervalo X, a resposta verdadeira do cálculo de uma função f para x, ou $f(x)$, estará contida em uma resposta obtida por uma extensão intervalar $F(X)$ da função $f(x)$.

3.2.10 Ferramentas Computacionais Intervalares

Existem diferentes plataformas em diferentes linguagens de programação com o intuito de implementar a aritmética intervalar nos computadores. Banerjee et al. (2015) afirma que com o passar dos anos e o desenvolvimento dessas plataformas computacionais, a possibilidade de analisar sistemas com incertezas paramétricas delimitadas como sistemas intervalares auxiliou na aplicação de técnicas de controle moderno para o projeto de sistemas de controle com modelos inexatos.

Neste trabalho, optou-se por trabalhar com o *Intlab*, um *toolbox* do *Matlab* caracterizado pela velocidade em que os cálculos são realizados (Rump, 1999).

Intlab

O *Intlab* é uma ferramenta designada para realizar operações com intervalos reais e complexos. O conceito do *Intlab* se divide em uma biblioteca rápida da aritmética intervalar e em um ambiente interativo de programação para operações intervalares de fácil acesso (Rump, 1999).

A entrada de um intervalo real no *Intlab* pode ser realizada através de quatro funções. A primeira a ser definida é a função `intval`, que apresenta duas variações.

A função `intval`(x_0) arredonda x_0 para o ponto flutuante mais próximo e então usa esse valor como os limites inferior e superior do intervalo. Com isso, se o número x_0 não é representado em ponto flutuante, implica que ele não estará contido no intervalo gerado. Rump (1999) afirma que isso é um problema de todas as bibliotecas desenvolvidas para as linguagens *Fortran 90*, *C++* ou *Matlab*.

Assim, é mais indicada a utilização de uma variação desta função, dada por `intval`('x_0'). Com o uso do símbolo ' ao redor da variável, o problema abordado anteriormente é contornado. Portanto, essa função cria um intervalo com os limites inferior e superior sendo, respectivamente, o ponto flutuante correspondente ao arredondamento negativo e o ponto flutuante correspondente ao arredondamento positivo, fazendo com que o número x_0 esteja contido no intervalo formado (Hargreaves, 2002).

A terceira função que define intervalos é a `midrad`(x_0,rad), que define um intervalo de ponto médio x_0 e raio rad, que define a metade do tamanho do intervalo (Rump, 1999).

Finalmente, na quarta função, o usuário entra com os pontos limites do intervalo ao invés de um ponto. A função é definida por `infsup`(inf,sup), onde os parâmetros inf e sup representam os limites inferior e superior, respectivamente. É importante salientar que se um número x_0 é arredondado para baixo e depois para cima, e esses valores servirem como parâmetros de entrada da função `infsup`, então será obtido o mesmo intervalo da função `intval`('x_0') (Hargreaves, 2002).

3.2.11 Conclusões da Seção

Enquanto a aritmética de ponto flutuante é afetada por erros de arredondamento, podendo produzir resultados imprecisos, a aritmética intervalar tem a vantagem de limitar a solução exata. A habilidade de alterar o modo de arredondamento em computadores foi um dos prin-

cipais fatores que facilitou a aplicação da aritmética intervalar em uma variedade de softwares computacionais (Hargreaves, 2002).

A análise intervalar utiliza intervalos fechados para representar as grandezas utilizadas. Contudo, é importante ressaltar que o grau de satisfação da solução é diretamente dependente ao tamanho do intervalo obtido. Sendo assim, o principal interesse na análise intervalar é obter resultados mais precisos possíveis, de modo que a precisão dessa solução seja alta.

Se os limites inferior e superior de um intervalo puderem ser definidos com o modo de arredondamento para baixo e para cima, respectivamente, então poderão ser feitos cálculos que não tem resultados afetados pela limitação de memória dos computadores utilizando intervalos, fazendo com que a solução desses cálculos seja limitada (Hargreaves, 2002).

3.3 Sistemas Dinâmicos Não-Lineares

Umas das áreas da ciência que aproveitou muito da capacidade de realizar cálculos dos computadores e com isso desenvolveu-se bastante foi a de sistemas dinâmicos, em especial, sistemas dinâmicos que apresentam comportamento caótico, sejam eles de tempo contínuo ou discreto (Hammel et al., 1987). Scheinerman (2011) afirma que qualquer sistema que evolui ao longo do tempo pode ser pensado como um sistema dinâmico.

Basicamente, os sistemas dinâmicos tentam entender, ou pelo menos descrever em equações matemáticas, mudanças ao longo do tempo que ocorrem em sistemas biológicos, químicos, econômicos, financeiros, eletrônicos, físicos ou artificiais (Lozi, 2013).

Um sistema dinâmico consiste em duas partes: um conjunto de estados, que descreve exatamente o estado de um sistema, e uma função ou regra, que, a partir do estado passado, determina o estado do sistema no próximo instante de tempo (Devaney, 1992).

O estudo de sistemas dinâmicos em que as leis de interação apresentam característica não linear oferece uma estrutura para o estudo e interpretação de muitos fenômenos naturais (Güémez e A., 1993).

3.3.1 Sistemas Dinâmicos Caóticos

Lorenz (1963) desenvolveu um modelo que simulava no computador a evolução das condições climáticas, a partir dos valores iniciais de ventos e temperaturas. A partir das simulações

feitas, foi possível constatar que pequenas modificações nas condições iniciais acarretariam grandes alterações na evolução do quadro como um todo, caracterizando um comportamento caótico. Com isso, esse trabalho veio a ser a base para o estudo do caos.

Sistemas dinâmicos não-lineares caóticos são objetos de análise da propagação de erros devido ao fato de que são definidos por apresentarem grande sensibilidade à condições iniciais, o que é conhecido como efeito borboleta. Isso significa que uma pequena diferença na condição inicial (tal como alguma diferença produzida por erro de arredondamento) pode produzir saídas totalmente divergentes, fazendo com que a predição de resposta em longo termo seja impossibilitada em sistemas caóticos (Lozi, 2013).

De fato, até os sistemas caóticos mais simples exibem um número de diferentes comportamentos que só podem ser totalmente analisados com a ajuda de hardware e software poderosos. Por isso, a análise de sistemas caóticos vem sendo um assunto clássico de pesquisa no campo da análise de erros (Tomida, 2008).

3.3.2 Sistemas Dinâmicos em Tempo Discreto

Sistemas dinâmicos podem ser em tempo contínuo ou tempo de discreto. Porém devido às dificuldades associadas com o estudo analítico de sistemas diferenciais, grande parte dos trabalhos tem se voltado a sistemas dinâmicos cujo estado é conhecido somente no tempo discreto (Gilmore e Lefranc, 2012).

Esses sistemas geralmente são definidos por uma função recursiva ou um mapa dado por:

$$x_{n+1} = f(x_n), \tag{3.34}$$

onde $n \in \mathbb{N}$, $M \subset \mathbb{R}$ é um espaço métrico e $f : M \to M$ é a função recursiva definida no espaço de estados e x_n denota o estado no tempo discreto n (Lozi, 2013).

A equação (3.34) pode ser escrita como um resultado de funções compostas, dado por (Nepomuceno, 2014):

$$x_n = f^1(x_{n-1}) = f^2(x_{n-2}) = \cdots = f^n(x_0) \tag{3.35}$$

Mapas unidimensionais são os sistemas mais simples capazes de apresentar comportamento caótico. Consequentemente, esses mapas são usados como ponto de partida no estudo do caos

(Tomida, 2008).

3.3.3 Órbitas e Pseudo-Órbitas

A sequência definida por $\{x_0,x_1,\ldots,x_n\}$ obtida a cada iteração do mapa da equação (3.34) é chamada órbita de x_0 (Gilmore e Lefranc, 2012).

No entanto, quando um computador é utilizado para calcular os resultados a cada iteração de um mapa, tem-se uma pseudo-órbita. Ou seja, o que se tem é uma aproximação da órbita verdadeira devido às propriedades inerentes aos computadores. Uma pseudo-órbita específica $i \in \mathbb{N}$ é representada por $\{\hat{x}_{i,0},\hat{x}_{i,1},\ldots,\hat{x}_{i,n}\}$, tal que

$$|x_n - \hat{x}_{i,n}| \leq \delta_{i,n} \tag{3.36}$$

onde $\delta_{i,n} \geq 0$ e $\delta_{i,n} \in \mathbb{R}$ (Nepomuceno e Martins, 2016).

Deve ser mencionado que pseudo-órbitas podem dar informações úteis a respeito do sistema devido a proximidade da orbita verdadeira. No entanto, isso é garantido apenas para sistemas em que o teorema do sombreamento é aplicável. Essa propriedade assegura que pseudo-órbitas permanecessem próximas a órbitas verdadeiras (Lozi, 2013). Devido ao fato desse teorema ser aplicado apenas em sistemas uniformemente hiperbólicos, em grande parte dos casos essa propriedade não se aplica.

3.3.4 Pontos Fixos e Órbitas Periódicas

Em sistemas dinâmicos discretos, os pontos fixos e as órbitas periódicas tem grande importância como elementos para determinação de pontos de equilíbrio destes sistemas (Devaney, 1992).

O ponto fixo de um mapa é um ponto para o qual $x_{n+1} = x_n = x^*$, ou ainda,

$$f(x^*) = x^*. \tag{3.37}$$

As órbitas periódicas ou pontos periódicos são uma generalização dos pontos fixos. Define-se que se $f^k(x^*) = x^*$, então tem-se uma órbita periódica de período k. Ou seja, os k pontos de uma órbita periódica se repetem a cada k iterações (Scheinerman, 2011).

3.4 Mapa Logístico

Um dos mapas unidimensionais mais estudados é o mapa logístico. Segundo Saha P. and Strogatz (1995), a equação do mapa logístico é estudada por ser acessível, exemplar, apresentar complexidade matemática e ser relevante para a ciência.

Robert May (1976) desenvolveu um sistema dinâmico de tempo discreto dado por

$$x_{n+1} = rx_n(1 - x_n). \tag{3.38}$$

Essa função, chamada de mapa logístico, descreve um modelo de flutuação populacional de moscas em um ambiente com quantidade de alimento constante (Lozi, 2013). Nesse modelo biológico, x_n é um número entre 0 e 1 que representa a razão entre a população existente e a maior possível na iteração n e r é a taxa de crescimento dessa população, que tem seus valores de interesse entre 0 e 4.

Segundo May (1976), para valores distintos do parâmetro r e da condição inicial x_0, a série pode apresentar comportamentos completamente diferentes, como equilíbrio, periodicidade e até caos. Esse fato leva à análise do comportamento da função de acordo com a variação de r através de um diagrama usado não só para o mapa logístico, mas também para outros, chamado de diagrama de bifurcação.

3.4.1 Diagrama de Bifurcação

O diagrama de bifurcação do mapa logístico (Fig. 3.2) mostra o comportamento do sistema da equação (3.38) para diferentes valores do parâmetro r. É importante observar que nem todas as condições iniciais levarão às respostas apresentadas no diagrama de bifurcação. Dependendo do valor de x_n, a órbita pode apresentar convergência a pontos fixos ou órbitas periódicas. Por exemplo, para $r = 4$ e $x_n = \frac{5-\sqrt{5}}{8}$, o sistema apresenta órbita periódica de período 2 como resposta (Lozi, 2013). E pode-se notar na Figura 3.2 que o comportamento representado no diagrama de bifurcação para esse valor de r é caótico.

Apesar de ser exaustivamente estudado, não se encontra na literatura um procedimento rigoroso para a construção do diagrama de bifurcação do mapa logístico (Paiva et al., 2015).

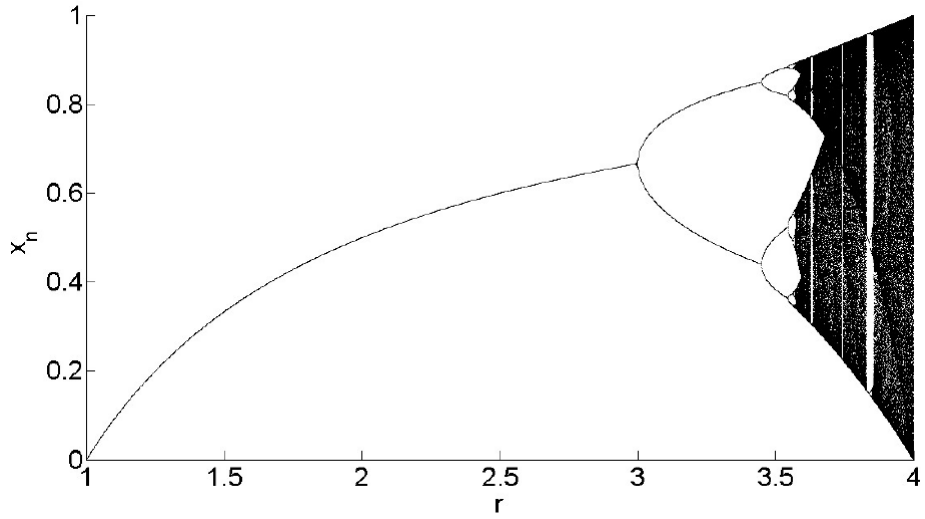


Figura 3.2: Diagrama de bifurcação do mapa logístico. Fonte: Monteiro (2006).

3.4.2 *Cobweb Plot*

O *cobweb plot* é outra ferramenta visual usada em sistemas dinâmicos para investigar funções recursivas unidimensionais (Paul et al., 2013). É um instrumento muito útil para monitorar o comportamento de um sistema dinâmico com a evolução de uma condição inicial sob várias iterações (Alpar, 2014).

O *cobweb plot*, mostrado na Figura 3.3, consiste de uma reta diagonal $y = x$ e uma curva representando o mapa $y = f(x)$. Para visualizar o comportamento do sistema a partir de uma condição inicial x_0 para um dado valor de parâmetro, deve-se seguir os seguintes passos (Tomida, 2008):

1. Traçar uma linha vertical do ponto x_0 até a curva da função. Isso levará à coordenada $(x_0,f(x_0))$.

2. Traçar uma linha horizontal do ponto $(x_0,f(x_0))$ até a reta diagonal $y = x$. Isso levará à coordenada $(f(x_0),f(x_0))$.

3. Traçar verticalmente do ponto $(f(x_0),f(x_0))$ para a curva da função, levando à coordenada $(f(x_0),f(f(x_0)))$.

4. Voltar ao passo 2 até atingir o número de iterações desejado.

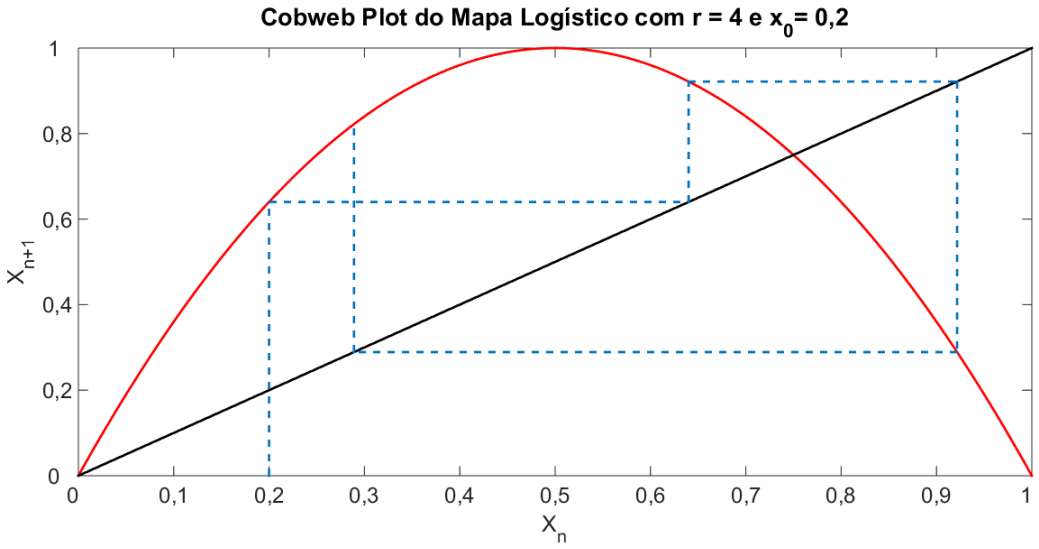


Figura 3.3: *Cobweb plot* do mapa logístico com $r = 4$ e $x_0 = 2$.

3.4.3 Comportamento do Mapa Logístico

Para todos os valores de r existem dois pontos fixos: $x = 0$, que é sempre instável e $x = \frac{r-1}{r}$, que é estável para $1 < r < 3$ e instável para $0 < r < 1$ e $3 < r < 4$. Porém, o comportamento do mapa logístico depende também da condição inicial x_0. Portanto, órbitas definidas sob um mesmo parâmetro r, podem apresentar comportamentos distintos (Lozi, 2013).

Diz-se que um ponto fixo x^* é estável se para todos os valores próximos a ele, ao decorrer das iterações, o sistema não só continua próximo mas tende a ele. Caso os valores permaneçam próximos ao ponto fixo, mas não convirjam à ele quando as iterações tendam a infinito, o ponto fixo é chamado marginalmente estável. Finalmente, o ponto fixo é chamado instável se não é estável nem marginalmente estável (Scheinerman, 2011).

Para valores de $0 < r < 3$, independente da condição inicial, o mapa logístico apresenta pontos fixos $x = \frac{r-1}{r}$ nas respostas. Para valores de $r > 3$, o sistema entra em uma região de comportamento periódico com bifurcações que representam duplicações de período. Esse comportamento periódico se mantém por $3 < r < r_\infty$, onde, segundo Feigenbaum (1978), $r_\infty \approx 3{,}569945672$. Para $r_\infty < r \leq 4$, o sistema geralmente apresenta comportamento caótico, exceto em casos onde são apresentadas janelas periódicas, também conhecidas como ilhas de estabilidade, que são representadas nessa região do diagrama de bifurcação (Fig. 3.2) por lacunas (Chaitin-Chatelin, 1996; Spandl, 2012).

Exemplo 3.7

Por exemplo, segundo Feigenbaum (1978), o mapa logístico para $r = 3{,}3$ e $x_0 = 0{,}75$ deve apresentar órbita periódica de período 2. Já para a mesma condição inicial e $r = 3{,}47$, a órbita deve convergir para período 4. Porém, mantendo a mesma condição inicial e fazendo $r = 4$, mesmo dentro de uma região caótica, a órbita convergirá para o ponto fixo, uma vez que $x_0 = \frac{r-1}{r}$.

CAPÍTULO 4

Metodologia

Foi desenvolvida uma metodologia intervalar voltada para a análise de mapas discretos, com uma aplicação direta na obtenção de pontos fixos e órbitas periódicas no mapa logístico. Basicamente, a metodologia consiste na junção de dois métodos intervalares, com intenção de reduzir ao máximo o erro obtido nas soluções a cada iteração. Os dois métodos serão melhores detalhados nas próximas seções.

4.1 Trabalhando com os Pontos Limites

O método intervalar proposto para solucionar o problema da influência da propagação do erro computacional em mapas discretos unidimensionais inicialmente consiste em produzir um intervalo a partir da condição inicial x_0, fazendo com que os limites $\underline{X_0}$ e $\overline{X_0}$ sejam os pontos flutuantes mais próximos de x_0.

Como já foi visto, a definição desses limites pode ser feita simplesmente arredondando a condição inicial x_0 para baixo e para cima, definindo então o intervalo $X_0 = [\underline{X_0},\overline{X_0}]$ em que $x_0 \in X_0$.

A partir do intervalo inicial, apenas os limites $\underline{X_0}$ e $\overline{X_0}$ são usados como variáveis nos cálculos seguintes. Portanto, ao invés de operações com todo o intervalo, através da aritmética intervalar tradicional (definida no Capítulo 3), os intervalos a cada iteração são calculados somente com os limites do intervalo da iteração passada.

Como será visto na subseção 4.1.2, é provado que trabalhar apenas com os limites dos intervalos, garante que todas as soluções do intervalo estejam contidas dentro das soluções dos limites desse intervalo para o mapa logístico.

Realizar operações aritméticas reais com os limites ao invés de operações aritméticas intervalares com todo o intervalo, diminui o efeito da superestimação, garantindo uma melhor precisão.

Além disso, cada operação realizada com os limites de um dado intervalo são arredondadas de modo que se garanta que o intervalo obtido na iteração seguinte abranja todas as possíveis respostas, assegurando que seu tamanho nunca seja nulo.

4.1.1 Arredondamento dos Resultados Obtidos

Através da análise do *cobweb plot* do mapa logístico (Fig. 3.3), pode-se observar que existem três situações de arredondamento para os resultados de cálculos com os limites intervalares:

- Limite superior $\overline{X_n}$ menor do que $0{,}5$: nesse caso, ambos os limites do intervalo estão na parte crescente da parábola do *cobweb plot*. Assim, para garantir que o intervalo abranja todos os possíveis resultados, $\underline{X_{n+1}}$ deve ser arredondado para baixo e $\overline{X_{n+1}}$ para cima.

- Limite inferior $\underline{X_n}$ maior do que $0{,}5$: quando o limite inferior do intervalo é maior do que $0{,}5$, indica que todo o intervalo está na parte decrescente do *cobweb plot*. Com isso, a solução encontrada com limite inferior $\underline{X_n}$ será o limite superior do intervalo na próxima iteração $\overline{X_{n+1}}$, e o inverso ocorre com o limite superior. Portanto, com a inversão dos limites, o arredondamento segue o mesmo padrão do primeiro caso, com $\underline{X_{n+1}}$ sendo arredondado para baixo e $\overline{X_{n+1}}$ para cima.

- Limite inferior $\underline{X_n}$ e superior $\overline{X_n}$ entre $0{,}5$: nesse caso, o ponto máximo da curva, dado por $f(0{,}5) = 0{,}25.r$, pertence ao intervalo X_{n+1}. Sendo assim, o limite superior da próxima iteração $\overline{X_{n+1}}$ deve receber esse ponto. O limite inferior da próxima iteração deve receber o menor dos dois resultados obtidos com os limites da iteração atual, ou seja, $\underline{X_{n+1}} = \min\{f(\underline{X_n}), f(\overline{X_n})\}$.

4.1.2 Aplicação do Método no Mapa Logístico

Nesta subseção será mostrado que, a cada iteração nos cálculos do mapa logístico com variável intervalar, as operações aritméticas com os intervalos podem ser substituídas por operações aritméticas reais dos seus limites, de modo que as soluções obtidas contenham toda a faixa de soluções dos valores que pertencem ao intervalo.

A demonstração será realizada por meio de teoremas que validam os cálculos com os limites para os três casos mostrados anteriormente. É importante lembrar que a equação do mapa logístico é dada por $x_{n+1} = f(x_n) = rx_n(1 - x_n)$, onde $0 < x_n < 1$ e $0 < r < 4$. Para a prova dos teoremas, é definido um número $\delta \in \mathbb{R}$ que auxiliará na representação dos demais números presentes no intervalo.

Caso 1: $\overline{X_n} < 0{,}5$

Será mostrado que a menor resposta possível de ser obtida para o mapa logístico dentro do intervalo $X_n = [\underline{X_n}, \overline{X_n}]$ é $f(\underline{X_n})$. Da mesma forma, $f(\overline{X_n})$ é a maior resposta a ser obtida dentro das respostas do intervalo X_n para uma dada iteração n. Isso também pode ser estabelecido por:

Teorema 4.1.1. *Se* $\overline{X_n} < 0{,}5$*, então* $f([\underline{X_n}, \overline{X_n}]) = [f(\underline{X_n}), f(\overline{X_n})]$.

Prova. Para que $f(\overline{X_n})$ seja o limite superior de $f([\underline{X_n}, \overline{X_n}])$, então $f(\overline{X_n}) > f(\overline{X_n} - \delta)$, para $0 < \delta < \overline{X_n}$. Tem-se que:

$$
\begin{aligned}
f(\overline{X_n} - \delta) &= r.(\overline{X_n} - \delta).[1 - (\overline{X_n} - \delta)], \\
f(\overline{X_n} - \delta) &= (r.\overline{X_n} - r.\delta).(1 - \overline{X_n} + \delta), \\
f(\overline{X_n} - \delta) &= r.\overline{X_n}.(1 - \overline{X_n}) + r.\overline{X_n}.\delta - r.\delta + r.\overline{X_n}.\delta - r.\delta.\delta, \\
f(\overline{X_n} - \delta) &= f(\overline{X_n}) + 2.r.\overline{X_n}.\delta - r.\delta - r.\delta.\delta.
\end{aligned}
$$

Para que $f(\overline{X_n}) > f(\overline{X_n} - \delta)$, então,

$$
\begin{aligned}
&2.r.\overline{X_n}.\delta - r.\delta - r.\delta.\delta < 0, \text{ ou ainda,} \\
&r.\delta.(2.\overline{X_n} - 1 - \delta) < 0. \text{ Como } r > 0 \text{ e } \delta > 0, \text{ então } r.\delta > 0. \text{ Logo,} \\
&2.\overline{X_n} - 1 - \delta < 0.
\end{aligned}
$$

Como $\overline{X_n} < 0{,}5$, então $2.\overline{X_n} - 1 < 0$. Assim, como $\delta > 0$, tem-se que $(2.\overline{X_n} - 1 - \delta) < 0$. Consequentemente, atesta-se que $f(\overline{X_n}) > f(\overline{X_n} - \delta)$ para $\overline{X_n} < 0{,}5$ e $0 < \delta < \overline{X_n}$.

Para que $f(\underline{X_n})$ seja o limite inferior de $f([\underline{X_n}, \overline{X_n}])$, então $f(\underline{X_n}) < f(\underline{X_n} + \delta)$, para $0 < \delta < 0{,}5 - \underline{X_n}$. Tem-se que:

$$f(\underline{X_n}+\delta) = r.(\underline{X_n}+\delta).[1-(\underline{X_n}+\delta)],$$
$$f(\underline{X_n}+\delta) = f(\underline{X_n}) + r.\delta - 2.r.\underline{X_n}.\delta - r.\delta.\delta,$$
$$f(\underline{X_n}+\delta) = f(\underline{X_n}) + r.\delta(1-2.\underline{X_n}-\delta).$$

Para que $f(\underline{X_n}) < f(\underline{X_n}+\delta)$, como $r.\delta > 0$, então, $(1-2.\underline{X_n}-\delta) > 0$. Admitindo o maior valor possível de δ, um valor muito próximo de $0{,}5 - \underline{X_n}$ é alcançado. Com isso,

$$1-2.\underline{X_n}-\delta > 0,$$
$$1-2.\underline{X_n}-(0{,}5-\underline{X_n}) > 0,$$
$$0{,}5-\underline{X_n} > 0.$$

Como $\underline{X_n} < 0{,}5$, então $0{,}5 - \underline{X_n} > 0$. Consequentemente, $f(\underline{X_n}) < f(\underline{X_n}+\delta)$, para $\overline{X_n} < 0{,}5$, estabelecendo então a prova para o Teorema 4.1.1. □

Caso 2: $\underline{X_n} > 0{,}5$

Esse é o caso de quando todo o intervalo reside na parte decrescente do *cobweb plot* do mapa logístico (Fig. 3.3). Como já foi dito, há uma inversão, ou seja, os limites inferior e superior da resposta na iteração $n+1$ serão obtidos pelos limites superior e inferior, respectivamente, da iteração n. Assim, assegura-se que:

Teorema 4.1.2. *Se* $\underline{X_n} > 0{,}5$*, então* $f([\underline{X_n},\overline{X_n}]) = [f(\overline{X_n}),f(\underline{X_n})]$.

Prova. Para que $f(\overline{X_n})$ seja o limite inferior de $f([\underline{X_n},\overline{X_n}])$, então $f(\overline{X_n}) < f(\overline{X_n}-\delta)$, onde $0 < \delta < \overline{X_n} - 0{,}5$. De maneira semelhante ao que já foi mostrado no Teorema 4.1.1, para que $f(\overline{X_n}) < f(\overline{X_n}-\delta)$, então $2.\overline{X_n} - 1 - \delta > 0$. Admitindo o maior valor possível de δ como algo próximo de $\overline{X_n} - 0{,}5$, tem-se que:

$$2.\overline{X_n} - 1 - \delta > 0,$$
$$2.\overline{X_n} - 1 - (\overline{X_n} - 0{,}5) > 0,$$
$$\overline{X_n} - 0{,}5 > 0.$$

Como $\overline{X_n} > 0{,}5$, então $\overline{X_n} - 0{,}5 > 0$. Consequentemente, $f(\overline{X_n}) < f(\overline{X_n}-\delta)$, para $\underline{X_n} > 0{,}5$ e $0 < \delta < \overline{X_n} - 0{,}5$.

Para que $f(\underline{X_n})$ seja o limite superior de $f([\underline{X_n},\overline{X_n}])$, então $f(\underline{X_n}) > f(\underline{X_n} + \delta)$, para $0 < \delta < 1 - \underline{X_n}$. Já foi mostrado que $f(\underline{X_n} + \delta) = f(\underline{X_n}) + r.\delta(1 - 2.\underline{X_n} - \delta)$. Portanto, como $r.\delta > 0$, para que $f(\underline{X_n}) > f(\underline{X_n} + \delta)$, então $1 - 2.\underline{X_n} - \delta < 0$. Como $\underline{X_n} > 0{,}5$ e $\delta > 0$, então $1 - 2.\underline{X_n} - \delta < 0$. Portanto, $f(\underline{X_n}) > f(\underline{X_n} + \delta)$, para $\underline{X_n} > 0{,}5$ e $0 < \delta < 1 - \underline{X_n}$, estabelecendo então a prova para o Teorema 4.1.2. □

Caso 3: $\underline{X_n} < 0{,}5 < \overline{X_n}$

Agora será analisado o caso em que o intervalo está tanto na parte crescente quanto decrescente do *cobweb plot* do mapa logístico (Fig. 3.3). Como o ponto máximo da curva pertence ao intervalo, então o limite superior da resposta na iteração seguinte recebe este ponto, dado por $f(0{,}5) = 0{,}25.r$. O limite inferior da próxima iteração recebe o menor dos valores obtidos com os limites do intervalo da iteração atual. Portanto, duas situações podem ser estabelecidas pelos seguintes teoremas:

Teorema 4.1.3. *Se* $\underline{X_n} < 0{,}5 < \overline{X_n}$ *e* $f(\underline{X_n}) < f(\overline{X_n})$*, então* $f([\underline{X_n},\overline{X_n}]) = [f(\underline{X_n}),f(0{,}5)]$.

Prova. Para que $f(\underline{X_n})$ seja o limite inferior de $f([\underline{X_n},\overline{X_n}])$, então $f(\underline{X_n}) < f(\underline{X_n} + \delta)$, onde $0 < \delta < 1 - 2.\underline{X_n}$. Como foi mostrado, para que $f(\underline{X_n}) < f(\underline{X_n} + \delta)$, então, como $r.\delta > 0$, tem-se que $1 - 2.\underline{X_n} - \delta > 0$. Como $\underline{X_n} < 0{,}5$, então $1 - 2.\underline{X_n} > 0$. E como $\delta < 1 - 2.\underline{X_n}$, tem-se que a condição é satisfeita, estabelecendo então a prova para o Teorema 4.1.3. □

Teorema 4.1.4. *Se* $\underline{X_n} < 0{,}5 < \overline{X_n}$ *e* $f(\overline{X_n}) < f(\underline{X_n})$*, então* $f([\underline{X_n},\overline{X_n}]) = [f(\overline{X_n}),f(0{,}5)]$.

Prova. Para que $f(\overline{X_n})$ seja o limite inferior de $f([\underline{X_n},\overline{X_n}])$, então $f(\overline{X_n}) < f(\overline{X_n} - \delta)$, onde $0 < \delta < 2.\overline{X_n} - 1$. Como foi mostrado, para que $f(\overline{X_n}) < f(\overline{X_n} - \delta)$, então, como $r.\delta > 0$, tem-se que $2.\overline{X_n} - 1 - \delta > 0$. Como $\overline{X_n} > 0{,}5$, então $2.\overline{X_n} - 1 > 0$. E como $\delta < 2.\overline{X_n} - 1$, tem-se que a condição é satisfeita, assegurando o Teorema 4.1.4. □

4.1.3 Analisando a Coerência do Método

A fim de verificar a coerência do método estudado, são calculadas, através do *Matlab*, as 10 primeiras iterações da pseudo-órbita do mapa logístico para $r = 3{,}6$ e $x_0 = 0{,}6$. Os resultados obtidos pelo *software*, que apresentam cerca de 16 dígitos de precisão, são comparados com cálculos simbólicos com precisão de 100 dígitos, obtidos pelo *software wxMaxima* 15.08.2.

O *wxMaxima* é um *software* gratuito que executa cálculos matemáticos, manipulações simbólicas e numéricas, além de criar gráficos no computador. Os cálculos foram feitos em um computador com processador *Intel i7-5500 @ 2.41GHz*, com *Windows* usado como sistema operacional.

A Tabela 4.1 apresenta os resultados simbólicos x_n obtidos pelo $wxMaxima$ e os intervalos $\underline{\hat{X}_n}$ e $\overline{\hat{X}_n}$ obtidos com a metodologia apresentada nesta seção.

Tabela 4.1: As 10 primeiras iterações obtidas como solução de $x_{n+1} = rx_n(1 - x_n)$, com $r = 3{,}6$ e $x_0 = 0{,}6$.

	wxMaxima	Aritmética Intervalar Desenvolvida	
n	x_n	$\underline{\hat{X}_n}$	$\overline{\hat{X}_n}$
0	0,600000000000000	0,600000000000000	0,600000000000000
1	0,864000000000000	0,864000000000000	0,864000000000000
2	0,423014400000000	0,423014399999999	0,423014400000001
3	0,878663582613504	0,878663582613503	0,878663582613505
4	0,383810008328301	0,383810008328300	0,383810008328303
5	0,851399589007190	0,851399589007189	0,851399589007192
6	0,455465983844079	0,455465983844076	0,455465983844083
7	0,892860197058087	0,892860197058086	0,892860197058088
8	0,344379116042931	0,344379116042928	0,344379116042935
9	0,812815705715114	0,812815705715110	0,812815705715119
10	0,547726803328638	0,547726803328628	0,547726803328647

Como se pode observar, todos os valores simbólicos calculados estão dentro do intervalo limitado pelos valores calculados de $\underline{X_n}$ e $\overline{X_n}$. Isso é uma indicação de que o método adotado apresenta coerência.

4.1.4 Convergindo para Pontos Fixos e Órbitas Periódicas

Como já visto no trabalho, convergir para pontos fixos ou órbitas periódicas consiste em encontrar pontos de estabilidade do sistema para determinadas condições iniciais e parâmetros adotados. Usando a análise intervalar, a convergência para esses pontos também pode ser observada. Através da interseção de dois intervalos, pode-se verificar se existem termos de um intervalo que pertencem também a outro. Quando dois intervalos, obtidos como respostas de um mapa em iterações diferentes, apresentam interseção diferente de conjunto vazio, existe a possibilidade de que estejam representando os mesmos dados. E a precisão dessa possibilidade depende do tamanho desses intervalos.

A utilização da técnica apresentada na seção anterior para o cálculo do mapa logístico defi-

nido sob um parâmetro r e uma dada condição inicial $X_0 = [\underline{X_0}, \overline{X_0}]$, resultará em uma pseudo-órbita intervalar representada pela sequência $\{\hat{X}_n\} = \{\hat{X}_0, \hat{X}_1, \ldots, \hat{X}_n\}$, onde cada intervalo da sequência é representado pelo limites usados nos cálculos.

A obtenção de pontos fixos e órbitas periódicas pode ser então definida como se segue.

Teorema 4.1.5 (Rodrigues Junior et al., 2016a)**.** *Seja a sequência $\{\hat{X}_0, \hat{X}_1, \ldots, \hat{X}_n\}$ a representação dos intervalos obtidos a cada iteração como resposta de um mapa recursivo. Se $X_n \cap X_{n+1} \neq \emptyset$, então $x^* \in X_n$ é ponto fixo.*

Prova. Por definição, se x^* é ponto fixo, então $f(x^*) = x^*$. Se $x^* \in X_n$, então $x^* \in f(X_n)$, uma vez que a resposta obtida pela aplicação da função sob o intervalo X_n compreende as respostas de todos os pontos pertencentes ao intervalo. Portanto, como $f(X_n) = X_{n+1}$, tem-se que $X_n \cap X_{n+1} \neq \emptyset$ e o ponto fixo x^* pertence ao intervalo obtido pela interseção dos dois intervalos. □

Teorema 4.1.6 (Rodrigues Junior et al., 2016a)**.** *Seja a sequência $\{\hat{X}_0, \hat{X}_1, \ldots, \hat{X}_n\}$ a representação dos intervalos obtidos a cada iteração como resposta de um mapa recursivo. Se $X_n \cap X_{n+p} \neq \emptyset$, então $x^* \in X_n$ caracteriza um dos pontos da órbita periódica de período p do mapa.*

Prova. Por definição, se x^* é um ponto pertencente a uma órbita periódica de período p, então $f^p(x^*) = x^*$. Se $x^* \in X_n$, então $x^* \in f^p(X_n)$. Portanto, como $f^p(X_n) = X_{n+p}$, tem-se que $X_n \cap X_{n+p} \neq \emptyset$ e x^* pertence ao intervalo obtido pela interseção dos dois intervalos. □

Portanto, a determinação do valor de x^* pode ser dada considerando o ponto médio do intervalo X_n com um erro associado dado pelo raio do intervalo, ou por $\omega(X_n)/2$. É importante salientar mais uma vez que a precisão da resposta depende do tamanho do intervalo. Com isso, pontos fixos e órbitas periódicas determinados com precisão baixa devem ser melhor investigados.

4.2 Reduzindo o Intervalo das Soluções

Nesta seção pretende-se utilizar os conceitos vistos no Capítulo 3 para diminuir a superestimação dos intervalos obtidos como resposta no cálculo de mapas discretos. A ideia principal

desta metodologia consiste em usar a interseção de diferentes pseudo-órbitas obtidas por extensões intervalares de uma função recursiva para reduzir os limites da solução exata dessa função.

Seja $\{x_n\} = \{x_0,x_1,\ldots,x_n\}$ a órbita verdadeira calculada para um mapa discreto unidimensional f e $\{\hat{X}_{i,n}\} = \{\hat{X}_{i,0},\hat{X}_{i,1},\ldots,\hat{X}_{i,n}\}$ a pseudo-órbita obtida computacionalmente por uma extensão intervalar de f, dada por F_i, com $i \in \mathbb{N}$. A partir da Definição 3.2.4 e pela dedução mostrada na seção 4.1.2, é claro que se $x_0 \in \hat{X}_{i,0}$, então

$$x_n \in \hat{X}_{i,n}. \tag{4.1}$$

A partir disso, pode-se estabelecer o que segue para duas extensões intervalares racionais F_1 e F_2 da mesma função f.

Teorema 4.2.1. *Se* $x_0 \in \hat{X}_{1,0}$ *e* $x_0 \in \hat{X}_{2,0}$*, então* $x_n \in [\hat{X}_{1,n} \cap \hat{X}_{2,n}]$*,* $n \in \mathbb{N}$*.*

Prova. Assumindo que $x_n \notin [\hat{X}_{1,n} \cap \hat{X}_{2,n}]$, tem-se que $x_n \notin \hat{X}_{1,n}$, ou $x_n \notin \hat{X}_{2,n}$, ou ambos os casos. De acordo com a equação (4.1), isso significa que $\hat{X}_{1,n}$ ou $\hat{X}_{2,n}$ não pertencem à pseudo-órbita obtida por uma extensão intervalar de f, gerando contradição. Isso completa a prova. □

Além disso, de acordo com a equação (3.14), se a interseção entre dois intervalos não é vazia, produz um novo intervalo com tamanho menor ou igual ao menor dos intervalos usados nessa operação algébrica. Portanto, como a interseção entre os intervalos obtidos por extensões intervalares de uma mesma função contém a resposta verdadeira, será verificado se essa interseção pode ser usada para reduzir os limites da órbita verdadeira.

4.3 Método Intervalar Desenvolvido

Com os métodos mostrados nas seções anteriores desse capítulo, pode-se obter a metodologia intervalar desenvolvida nessa dissertação, que consiste basicamente na aplicação desses métodos para análise e cálculo de pontos fixos e órbitas periódicas de mapas recursivos. Portanto, a metodologia proposta consiste em duas abordagens: a primeira corresponde à aplicação do método descrito na seção 4.1, que trabalha com os pontos limites dos intervalos obtidos como resposta por uma dada extensão intervalar; a segunda é a junção dos métodos descritos

nas seções 4.1 e 4.2, que busca aumentar a precisão das respostas através da interseção entre pseudo-órbitas obtidas por extensões intervalares distintas. A seguir, são mostrados os passos para implementação do segundo método em um mapa recursivo unidimensional:

1. Obter extensões intervalares F_i da função do mapa f a ser estudado.

2. Definir os mesmos parâmetros para todas as funções intervalares e um número máximo de iterações n.

3. Alterando o modo de arredondamento, definir um intervalo $\hat{X}_{i,0} = [\underline{\hat{X}_{i,0}}, \overline{\hat{X}_{i,0}}]$ correspondente a extensão intervalar, de modo que a condição inicial $x_0 \in \hat{X}_0$, e $\underline{\hat{X}_{i,0}}$ e $\overline{\hat{X}_{i,0}}$ sejam o menor e maior pontos flutuantes mais próximos de x_0, respectivamente.

4. Calcular a resposta da iteração atual para cada extensão intervalar. Isso deve ser feito com o cálculo de cada limite na função do mapa e com o arredondamento dos resultados, de acordo com a explicação da seção 4.1.

5. Calcular a interseção entre os intervalos obtidos para cada extensão intervalar, de modo a obter um intervalo no máximo do mesmo tamanho do menor dos dois intervalos. O intervalo obtido nesse passo, deve representar o intervalo da pseudo-órbita da função para a iteração atual.

6. Verificar a ocorrência de ponto fixo ou órbita periódica de acordo com os Teoremas 4.1.5 e 4.1.6. Se a pseudo-órbita calculada atingir alguma região de estabilidade, deve-se então parar os cálculos e realizar uma análise dos intervalos obtidos.

7. Voltar ao passo 4 e repetir os cálculos usando o intervalo obtido com a interseção entre as respostas das extensões intervalares na iteração atual até que o número de iterações n seja alcançado ou a pseudo-órbita tenha convergido para uma região estável.

A metodologia descrita na seção 4.1 pode ser estabelecida por um esquema similar ao apresentado. Basta que o processo seja feito para apenas uma extensão intervalar, de modo que o passo 5 seja eliminado.

4.4 Propostas

Para avaliar a metodologia intervalar desenvolvida, será feita a análise do cálculo de pontos fixos e órbitas periódicas do mapa logístico, dado pela equação (3.38). Para aplicar os métodos abordados, serão utilizadas duas extensões intervalares do mapa logístico, dadas por:

$$F_1(X_n) = rX_n(1 - X_n)$$

$$F_2(X_n) = r(X_n(1 - X_n))$$

Assim, o método descrito na seção 4.1 irá calcular separadamente os resultados obtidos por cada extensão e o método que usará a interseção entre as pseudo-órbitas obtidas por $F_1(X_n)$ e $F_2(X_n)$ será denotado por $F_1 \cap F_2$. Como já mencionado, nesse método a interseção é realizada a cada iteração e o intervalo obtido será a resposta usada como entrada na próxima iteração.

Além de serem comparados entre si, esses resultados serão confrontados com o cálculo intervalar realizado pelo *Intlab* e pelo método desenvolvido em Nepomuceno (2014), descrito na seção 2.2. Todos os métodos serão avaliados de acordo com os tamanhos dos intervalos finais obtidos, o número de iterações e a convergência para pontos fixos ou órbitas periódicas alcançados de acordo com resultados presentes na literatura.

Serão calculadas órbitas periódicas das primeiras 11 bifurcações do mapa logístico. Como já mencionado no Capítulo 3, essa região ocorre entre $3 < r < r_\infty$, onde $r_\infty \approx 3{,}569945672$. A Tabela 4.2, baseada em Rasband (2015) e Feigenbaum (1978), apresenta os pontos de duplicação de período para esse mapa.

Além das órbitas periódicas, limitadas pelos valores de r representados na Tabela 4.2, também será analisada a convergência para ponto fixo a partir dos valores de $r = \frac{327}{100}$ e $x_0 = \frac{100}{327}$, mostrado analiticamente no Capítulo 1. Portanto 12 casos de convergência para ponto fixo ou órbitas periódicas serão calculados e comparados:

- **Caso 1**: $r = \frac{327}{100}$ e $x_0 = \frac{100}{327}$, cuja resposta deveria convergir para ponto fixo;
- **Caso 2**: $r = 3{,}3$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 2;

Tabela 4.2: Pontos de duplicação de período para o mapa logístico de acordo com o parâmetro r. Fonte: Rasband (2015).

Período	r
2	3,0
4	3,449490000
8	3,544090000
16	3,564407000
32	3,568759000
64	3,569692000
128	3,569891000
256	3,569934000
512	3,569943000
1024	3,569945100
2048	3,569945557
r_∞	3,569945672

- **Caso 3**: $r = 3{,}47$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 4;
- **Caso 4**: $r = 3{,}55$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 8;
- **Caso 5**: $r = 3{,}566$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 16;
- **Caso 6**: $r = 3{,}5689$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 32;
- **Caso 7**: $r = 3{,}5698$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 64;
- **Caso 8**: $r = 3{,}5699$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 128;
- **Caso 9**: $r = 3{,}56994$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 256;
- **Caso 10**: $r = 3{,}5699440$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 512;
- **Caso 11**: $r = 3{,}5699453$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 1024;

- **Caso 12**: $r = 3{,}5699456$ e $x_0 = 0{,}6$, cuja resposta deveria convergir para órbita periódica de período 2048;

4.5 *Hardware* e *Software* Utilizados

Todos os cálculos foram realizados através do *software Matlab* em um computador com processador *Intel i7-5500 @ 2.41GHz*, com sistema operacional *Windows*. Além disso, as rotinas usadas para implementar as metodologias desenvolvidas nessa dissertação estão disponíveis no final do documento, na seção Apêndice.

CAPÍTULO 5

Resultados

Os resultados da aplicação das metodologias abordadas no trabalho, assim como dos métodos intervalares resultantes pelo emprego do *Intlab* e da técnica de Nepomuceno (2014) serão apresentados em tabelas ao longo deste capítulo.

As Tabelas 5.1 a 5.12 apresentam os cálculos dos 12 casos abordados no capítulo anterior para cada uma das metodologias. Em cada método, as órbitas foram calculadas para as extensões intervalares $F_1(x_n) = rx_n(1 - x_n)$ e $F_2(x_n) = r(x_n(1 - x_n))$. As tabelas apresentam o período da órbita alcançado, o tamanho do intervalo final obtido e o número de iterações necessárias para atingir o ponto fixo ou a órbita periódica.

Tabela 5.1: Comparação dos cálculos para o Caso 1 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	1	$3{,}3307 \times 10^{-16}$	2
	F_2	1	$3{,}3307 \times 10^{-16}$	2
	$F_1 \cap F_2$	1	$3{,}3307 \times 10^{-16}$	2
Nepomuceno (2014)	F_1	1	$1{,}6527 \times 10^{-15}$	2
	F_2	1	$1{,}0474 \times 10^{-15}$	2
Intlab	F_1	1	$2{,}4425 \times 10^{-15}$	2
	F_2	1	$2{,}4425 \times 10^{-15}$	2

Pela análise da Tabela 5.1, pode-se perceber que todos os métodos necessitaram do mesmo número de iterações para convergir para o ponto fixo. Nota-se uma pequena diferença no tamanho dos intervalos entre os três métodos abordados. Dentro de cada metodologia, os tamanhos dos intervalos tendem a ser os mesmos e pequenos devido ao fato da resposta convergir já na segunda iteração para o ponto fixo, apresentando a mesma solução do cálculo feito analiticamente no Capítulo 1.

Já para o segundo caso, mostrado na Tabela 5.2, apesar de todos os métodos convergirem para órbitas com o período esperado, existem diferenças significativas em relação à quantidade de iterações e ao tamanho final do intervalo encontrado. Percebe-se que, apesar de necessitarem de um número muito maior de iterações em relação aos outros métodos, as metodologias desenvolvidas nessa dissertação apresentam intervalos com precisão muito maior.

Tabela 5.2: Comparação dos cálculos para o Caso 2 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	2	$6{,}6613 \times 10^{-16}$	55
	F_2	2	$4{,}4409 \times 10^{-16}$	55
	$F_1 \cap F_2$	2	$4{,}4409 \times 10^{-16}$	55
Nepomuceno (2014)	F_1	2	$1{,}4601 \times 10^{-8}$	27
	F_2	2	$2{,}4474 \times 10^{-8}$	28
Intlab	F_1	2	$3{,}6590 \times 10^{-6}$	20
	F_2	2	$3{,}6590 \times 10^{-6}$	20

Para os Casos 3 e 4, cuja órbita deveria convergir para períodos 4 e 8, respectivamente, as respostas, mostradas nas Tabelas 5.3 e 5.4, apresentam comportamento parecido com o caso anterior: todos os métodos convergem para órbitas com o período esperado, embora haja uma diferença considerável no tamanho final dos intervalos e no número de iterações da metodologia desenvolvida em relação aos demais métodos.

Tabela 5.3: Comparação dos cálculos para o Caso 3 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	4	$1{,}1102 \times 10^{-15}$	207
	F_2	4	$8{,}8818 \times 10^{-16}$	207
	$F_1 \cap F_2$	4	$8{,}8818 \times 10^{-16}$	207
Nepomuceno (2014)	F_1	4	$7{,}4941 \times 10^{-5}$	33
	F_2	4	$3{,}2963 \times 10^{-5}$	33
Intlab	F_1	4	$6{,}4472 \times 10^{-4}$	23
	F_2	4	$6{,}4472 \times 10^{-4}$	23

As respostas obtidas com o *Intlab* para o Caso 5, ao invés de atingirem período 16, como era esperado, convergiram para órbitas com período 8. Pode-se notar que o tamanho dos intervalos obtidos para esse método é bem maior em relação aos demais, mostrando que as respostas obtidas apresentam precisão mais baixa. Todos os outros métodos convergiram para órbitas com o período esperado para ambas as extensões, como se pode observar na Tabela 5.5.

Para o sexto caso, apresentado na Tabela 5.6, apenas a metodologia desenvolvida neste

Tabela 5.4: Comparação dos cálculos para o Caso 4 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	8	$1{,}6653 \times 10^{-15}$	299
	F_2	8	$6{,}6613 \times 10^{-16}$	303
	$F_1 \cap F_2$	8	$5{,}5511 \times 10^{-16}$	303
Nepomuceno (2014)	F_1	8	$2{,}1521 \times 10^{-3}$	35
	F_2	8	$9{,}4794 \times 10^{-4}$	35
Intlab	F_1	8	$9{,}1500 \times 10^{-4}$	23
	F_2	8	$9{,}1500 \times 10^{-4}$	23

Tabela 5.5: Comparação dos cálculos para o Caso 5 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	16	$6{,}6613 \times 10^{-16}$	415
	F_2	16	$2{,}1094 \times 10^{-15}$	417
	$F_1 \cap F_2$	16	$5{,}5511 \times 10^{-16}$	415
Nepomuceno (2014)	F_1	16	$3{,}9800 \times 10^{-4}$	32
	F_2	16	$3{,}6663 \times 10^{-4}$	33
Intlab	F_1	8	0,0012	23
	F_2	8	0,0012	23

trabalho apresentou resultados de acordo com a literatura. A órbita encontrada pelo método que usa a ferramenta *Intlab* continuou com período 8 e o método de Nepomuceno (2014) apresentou período 16, ao invés de convergirem para período 32. Mais uma vez, nota-se que a quantidade de iterações da metodologia desenvolvida é muito maior em relação aos demais métodos, porém o tamanho do intervalo final permanece pequeno.

Tabela 5.6: Comparação dos cálculos para o Caso 6 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	32	$1{,}1546 \times 10^{-14}$	1991
	F_2	32	$6{,}4393 \times 10^{-15}$	1999
	$F_1 \cap F_2$	32	$3{,}1086 \times 10^{-15}$	2015
Nepomuceno (2014)	F_1	16	0,0028	34
	F_2	16	0,0018	35
Intlab	F_1	8	0,0110	25
	F_2	8	0,0110	25

Os demais casos estão representados nas Tabelas 5.7 a 5.12. Nota-se que a metodologia desenvolvida neste trabalho converge para órbitas com os períodos esperados de acordo com a literatura em todos os casos. Além disso, é possível observar que os demais métodos não conseguem alcançar órbitas com esses períodos e estacionam suas respostas em período 8, no

caso do método intervalar que usa *Intlab*, e 16, no caso do método abordado por Nepomuceno (2014).

Um fato interessante que não havia ocorrido para os casos anteriores é que, a partir do Caso 7, o tamanho do intervalo final obtido para o método que usa a interseção das respostas entre as extensões intervalares é menor em relação ao tamanho apresentado para uma ou todas as extensões quando aplicadas no método desenvolvido. Isso mostra que, apesar de obter intervalos muito pequenos, o método da interseção entre as respostas das extensões intervalares a cada iteração não reduz os limites da solução verdadeira para todos os casos.

Tabela 5.7: Comparação dos cálculos para o Caso 7 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	64	$8{,}1046 \times 10^{-15}$	809
	F_2	64	$5{,}2180 \times 10^{-15}$	811
	$F_1 \cap F_2$	64	$3{,}8969 \times 10^{-14}$	829
Nepomuceno (2014)	F_1	16	$4{,}4477 \times 10^{-4}$	32
	F_2	16	0,0013	34
Intlab	F_1	8	0,0134	25
	F_2	8	0,0134	25

Tabela 5.8: Comparação dos cálculos para o Caso 8 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	128	$2{,}5313 \times 10^{-14}$	4351
	F_2	128	$3{,}9968 \times 10^{-15}$	4383
	$F_1 \cap F_2$	128	$1{,}1546 \times 10^{-14}$	4391
Nepomuceno (2014)	F_1	16	$4{,}4737 \times 10^{-4}$	32
	F_2	16	$4{,}1602 \times 10^{-4}$	33
Intlab	F_1	8	0,0116	25
	F_2	8	0,0116	25

Apesar de necessitarem de números de iterações muito grandes, os intervalos finais obtidos na metodologia desenvolvida são pequenos em relação à precisão da máquina. O maior intervalo encontrado foi da ordem de $2{,}4 \times 10^{-13}$ para o método $F_1 \cap F_2$ no Caso 11, e esse mesmo método necessitou de 21887 iterações para convergir para a órbita periódica do Caso 12.

A análise dos resultados das Tabelas 5.7 a 5.12 pode gerar o questionamento das propriedades que foram apresentadas ao longo desta dissertação. Isso pelo fato de que o método $F_1 \cap F_2$ apresentou nesses casos intervalos finais com tamanhos maiores do que os cálculos convencionais pelas extensões intervalares. Porém, a interseção das respostas de F_1 e F_2 só garante

Tabela 5.9: Comparação dos cálculos para o Caso 9 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	256	$1{,}2434 \times 10^{-14}$	4517
	F_2	256	$1{,}4100 \times 10^{-14}$	4521
	$F_1 \cap F_2$	256	$3{,}3973 \times 10^{-14}$	4533
Nepomuceno (2014)	F_1	16	$4{,}4842 \times 10^{-4}$	32
	F_2	16	$4{,}1701 \times 10^{-4}$	33
Intlab	F_1	8	0,0133	25
	F_2	8	0,0133	25

Tabela 5.10: Comparação dos cálculos para o Caso 10 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	512	$1{,}2212 \times 10^{-15}$	3999
	F_2	512	$3{,}9968 \times 10^{-15}$	4001
	$F_1 \cap F_2$	512	$4{,}1078 \times 10^{-15}$	4003
Nepomuceno (2014)	F_1	16	$4{,}4852 \times 10^{-4}$	32
	F_2	16	$4{,}1711 \times 10^{-4}$	33
Intlab	F_1	8	0,01558	25
	F_2	8	0,01558	25

Tabela 5.11: Comparação dos cálculos para o Caso 11 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	1024	$6{,}3616 \times 10^{-14}$	10711
	F_2	1024	$9{,}4480 \times 10^{-14}$	10739
	$F_1 \cap F_2$	1024	$2{,}3948 \times 10^{-13}$	10833
Nepomuceno (2014)	F_1	16	$4{,}4856 \times 10^{-4}$	32
	F_2	16	$4{,}1715 \times 10^{-4}$	33
Intlab	F_1	8	0,0116	25
	F_2	8	0,0116	25

Tabela 5.12: Comparação dos cálculos para o Caso 12 entre as metodologias desenvolvidas, *Intlab* e Nepomuceno (2014).

		Período	Tamanho do Intervalo	Iterações
Metodologia Desenvolvida	F_1	2048	$8{,}9040 \times 10^{-14}$	21551
	F_2	2048	$3{,}9524 \times 10^{-14}$	21695
	$F_1 \cap F_2$	2048	$2{,}2249 \times 10^{-13}$	21887
Nepomuceno (2014)	F_1	16	$4{,}4857 \times 10^{-4}$	32
	F_2	16	$4{,}1715 \times 10^{-4}$	33
Intlab	F_1	8	0,0134	25
	F_2	8	0,0134	25

que os limites do resultado final sejam reduzidos se a interseção for realizada após o cálculo das pseudo-órbitas, ou simplesmente, se a interseção realizada em uma dada iteração não seja usada como entrada da próxima.

Isso não afeta os resultados obtidos, pois como foi mostrado, a órbita verdadeira estará sempre contida tanto nas pseudo-órbitas obtidas por F_1 e F_2, quanto pela pseudo-órbita resultante de $F_1 \cap F_2$ a cada iteração.

A interseção ao final do cálculo das pseudo-órbitas de F_1 e F_2 poderia até reduzir o tamanho do intervalo obtido em alguns casos. Mas usar esse método faz com que sua resposta convirja obrigatoriamente para órbitas com os mesmos períodos das obtidas pelas extensões intervalares. Além disso, caso F_1 e F_2 convergissem para órbitas com períodos distintos, ou apresentassem divergência em suas respostas, a resposta obtida pela interseção entre essas pseudo-órbitas também seria prejudicada.

Por isso, como o método $F_1 \cap F_2$ pode apresentar respostas mais independentes das respostas convencionais obtidas pelas extensões intervalares, optou-se por realizar a interseção a cada iteração.

Aprofundando na metodologia desenvolvida nessa dissertação, a Tabela 5.13 traz a média e o tamanho máximo dos intervalos obtidos para F_1, F_2 e $F_1 \cap F_2$ nos 12 casos já abordados.

Pode-se notar que, mesmo o método $F_1 \cap F_2$ apresentando tamanhos de intervalos finais maiores para alguns casos, tanto os valores médios quanto os maiores intervalos obtidos para esse método são menores em todos os casos em relação aos resultados apresentados por F_1 e F_2.

Além disso, pode ser feita uma observação em relação à extensão intervalar F_2, que apresenta tamanho médio de intervalos menor que F_1 em todos os casos. Isso, aliado ao fato de que a extensão F_2 apresenta os intervalos finais (Tab. 5.1 a 5.12) e os maiores intervalos (Tab. 5.13) menores em relação a F_2 na maioria dos casos, pode ser uma sugestão de que a extensão $F_2(x_n) = r(x_n.(1 - x_n))$ possa ter prioridade em relação à $F_1(x_n) = rx_n.(1 - x_n)$ quando se trabalha com a análise intervalar do mapa logístico.

Um novo questionamento geralmente é feito com relação ao tamanho máximo dos intervalos: por que o maior intervalo obtido em uma pseudo-órbita não é o intervalo final, visto que, ao longo das iterações, os intervalos tendem a aumentar de tamanho? A afirmação de que o tamanho dos intervalos deve aumentar a cada iteração não é satisfeita no caso do mapa logístico, quando aplicada a metodologia desenvolvida neste trabalho.

Isso ocorre pois essa metodologia trabalha com os limites dos intervalos, evitando as operações intervalares, que superestimam os intervalos. A afirmação do parágrafo anterior pode ser

Tabela 5.13: Média de tamanho dos intervalos ao longo dos cálculos e tamanho final dos intervalos obtidos com a metodologia desenvolvida para F_1, F_2 e $F_1 \cap F_2$.

		Tamanho Médio dos Intervalos	Maior Intervalo
Caso 1	F_1	$1{,}6653 \times 10^{-16}$	$3{,}3307 \times 10^{-16}$
	F_2	$1{,}6653 \times 10^{-16}$	$3{,}3307 \times 10^{-16}$
	$F_1 \cap F_2$	$1{,}6653 \times 10^{-16}$	$3{,}3307 \times 10^{-16}$
Caso 2	F_1	$7{,}5832 \times 10^{-16}$	$1{,}3323 \times 10^{-15}$
	F_2	$7{,}2462 \times 10^{-16}$	$1{,}6653 \times 10^{-15}$
	$F_1 \cap F_2$	$6{,}1260 \times 10^{-16}$	$1{,}3323 \times 10^{-15}$
Caso 3	F_1	$2{,}8481 \times 10^{-15}$	$6{,}8834 \times 10^{-15}$
	F_2	$2{,}7526 \times 10^{-15}$	$6{,}1062 \times 10^{-15}$
	$F_1 \cap F_2$	$2{,}5071 \times 10^{-15}$	$5{,}5511 \times 10^{-15}$
Caso 4	F_1	$4{,}7772 \times 10^{-15}$	$2{,}0317 \times 10^{-14}$
	F_2	$4{,}7425 \times 10^{-15}$	$2{,}0983 \times 10^{-14}$
	$F_1 \cap F_2$	$3{,}9917 \times 10^{-15}$	$1{,}7430 \times 10^{-14}$
Caso 5	F_1	$1{,}2874 \times 10^{-14}$	$1{,}0225 \times 10^{-13}$
	F_2	$1{,}1226 \times 10^{-14}$	$8{,}9539 \times 10^{-14}$
	$F_1 \cap F_2$	$1{,}0531 \times 10^{-14}$	$8{,}1657 \times 10^{-14}$
Caso 6	F_1	$5{,}1288 \times 10^{-14}$	$2{,}7700 \times 10^{-13}$
	F_2	$4{,}7384 \times 10^{-14}$	$2{,}5546 \times 10^{-13}$
	$F_1 \cap F_2$	$4{,}2141 \times 10^{-14}$	$2{,}3392 \times 10^{-13}$
Caso 7	F_1	$1{,}1282 \times 10^{-13}$	$2{,}2962 \times 10^{-12}$
	F_2	$1{,}0768 \times 10^{-13}$	$2{,}2191 \times 10^{-12}$
	$F_1 \cap F_2$	$9{,}6023 \times 10^{-14}$	$2{,}0486 \times 10^{-12}$
Caso 8	F_1	$2{,}3601 \times 10^{-13}$	$2{,}4866 \times 10^{-12}$
	F_2	$2{,}1823 \times 10^{-13}$	$2{,}3040 \times 10^{-12}$
	$F_1 \cap F_2$	$1{,}9269 \times 10^{-13}$	$2{,}0218 \times 10^{-12}$
Caso 9	F_1	$7{,}6986 \times 10^{-13}$	$4{,}0730 \times 10^{-11}$
	F_2	$7{,}1601 \times 10^{-13}$	$3{,}8006 \times 10^{-11}$
	$F_1 \cap F_2$	$6{,}2977 \times 10^{-13}$	$3{,}3817 \times 10^{-11}$
Caso 10	F_1	$6{,}8030 \times 10^{-13}$	$1{,}5406 \times 10^{-11}$
	F_2	$6{,}3138 \times 10^{-13}$	$1{,}4120 \times 10^{-11}$
	$F_1 \cap F_2$	$5{,}5653 \times 10^{-13}$	$1{,}2714 \times 10^{-11}$
Caso 11	F_1	$2{,}1592 \times 10^{-12}$	$9{,}4804 \times 10^{-11}$
	F_2	$1{,}9994 \times 10^{-12}$	$8{,}8857 \times 10^{-11}$
	$F_1 \cap F_2$	$1{,}7773 \times 10^{-12}$	$7{,}8888 \times 10^{-11}$
Caso 12	F_1	$8{,}2942 \times 10^{-11}$	$7{,}4813 \times 10^{-9}$
	F_2	$7{,}5437 \times 10^{-11}$	$6{,}8016 \times 10^{-9}$
	$F_1 \cap F_2$	$6{,}5823 \times 10^{-11}$	$5{,}9891 \times 10^{-9}$

facilmente mostrada através do *cobweb plot* do mapa logístico. Intervalos em zonas de maior inclinação tendem a apresentar respostas maiores em relação a intervalos de mesmo tamanho que estejam em regiões com inclinação menor.

A Figura 5.1 apresenta um esquema representando o que foi afirmado, onde os intervalos X_A e X_B, representados no gráfico pelas áreas vermelhas no eixo x, possuem o mesmo tamanho na iteração n. Quando os limites desses intervalos são usados para calcular a resposta da

próxima iteração, produzem intervalos com tamanhos distintos. Como se pode ver na imagem, o tamanho das respostas correspondentes a cada intervalo na iteração $n + 1$, representado pela área coberta no eixo y, é muito maior para o intervalo X_A. Além disso, a resposta de X_B tem tamanho menor que o próprio intervalo, mostrando que a afirmação de que os intervalos devem crescer ao longo das iterações nem sempre é satisfeita.

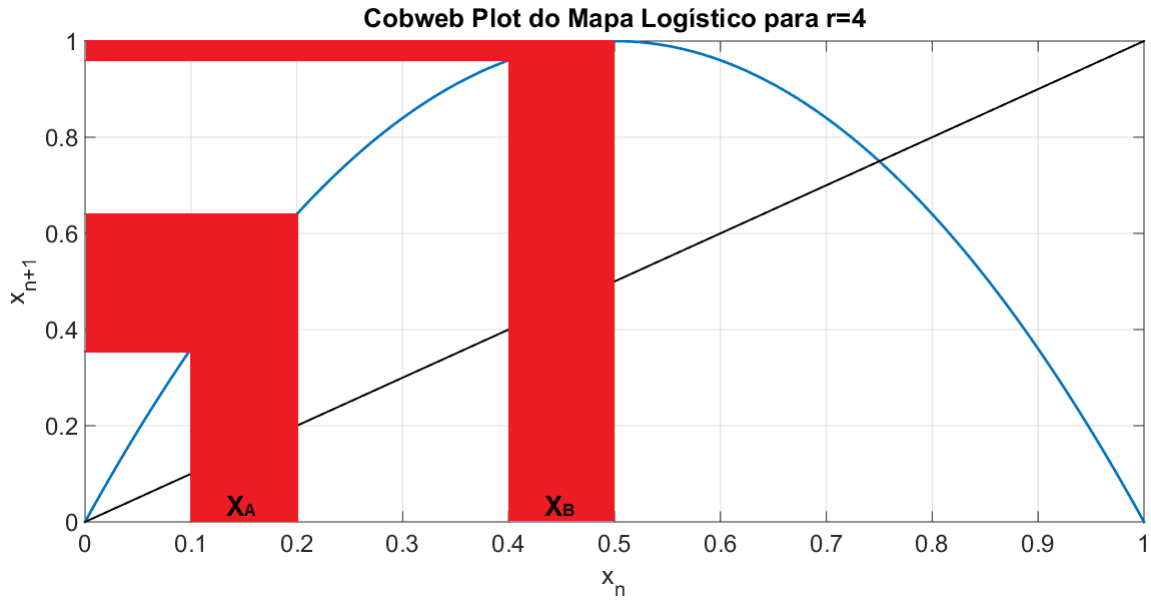


Figura 5.1: Evolução nos tamanhos de dois intervalos com o mesmo tamanho par o mapa logístico.

Um outro fator que influencia no crescimento dos intervalos ao longo das iterações no cálculo do mapa logístico é o parâmetro r, que determina a inclinação da curva do *cobweb plot*. Sendo assim, quanto maior o parâmetro r, maior a tendência de crescimento dos intervalos.

Portanto, a inclinação da curva e a posição do intervalo determinam o tamanho dos intervalos obtidos. Por isso, pode existir uma variação entre crescimento, estabilização e decrescimento no tamanho dos intervalos a cada iteração, o que mostra que o intervalo com tamanho máximo não é, necessariamente, o intervalo final.

Para uma melhor visualização, a Figura 5.2 mostra a evolução nos tamanhos dos intervalos para os cálculos de F_1, F_2 e $F_1 \cap F_2$ na metodologia desenvolvida para o Caso 7. Pode-se notar que os intervalos obtidos pelos três cálculos seguem o mesmo padrão de evolução de acordo com as iterações passadas.

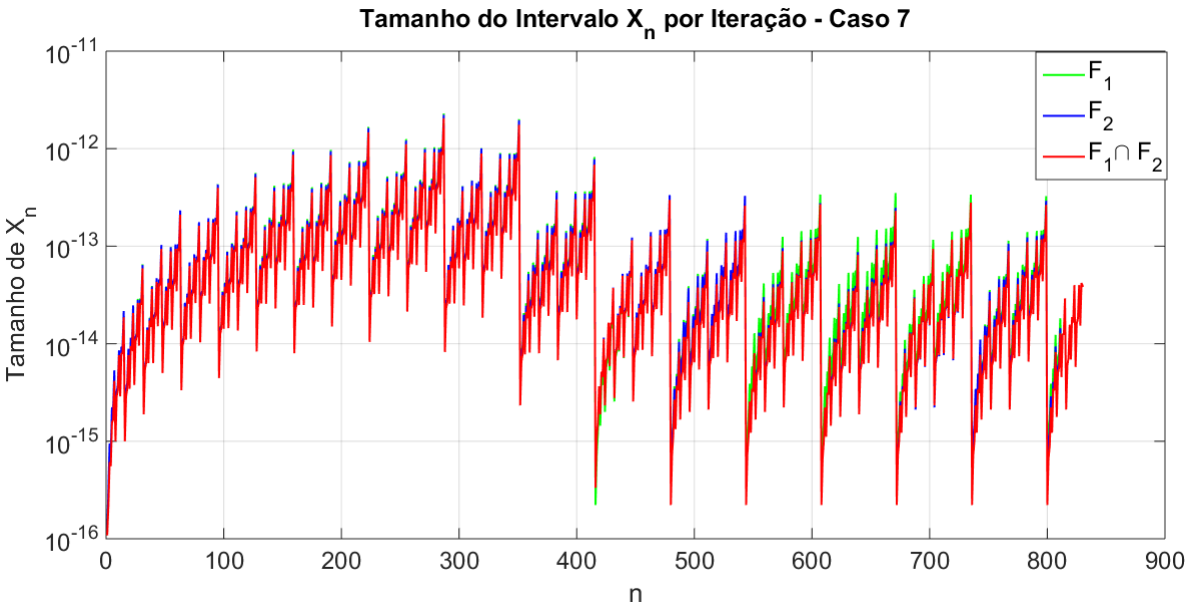


Figura 5.2: Evolução nos tamanhos dos intervalos para F_1, F_2 e $F_1 \cap F_2$ na metodologia desenvolvida para o Caso 7.

CAPÍTULO 6

Conclusões

6.1 Considerações Finais

Neste trabalho foi desenvolvida uma metodologia intervalar para análise de pontos fixos e órbitas periódicas no mapa logístico, que pode ser estendida para outros mapas discretos unidimensionais, desde que satisfaçam as condições estipuladas. A metodologia consiste em trabalhar com intervalos gerados a partir do arredondamento da condição inicial x_0 para os pontos flutuantes mais próximos, de modo que o problema da influência da propagação do erro em cálculos computacionais seja limitado.

Nesta metodologia, a partir da criação do intervalo baseado na condição inicial x_0, garante-se que todas as possibilidades de respostas correspondentes aos valores limitados pela faixa do intervalo sejam obtidas a cada iteração. Isso faz com que a resposta verdadeira também esteja dentro dos intervalos gerados.

Em muitos casos, cálculos da órbita de um mapa são computados sem análise da propagação de erros, o que pode gerar resultados pouco confiáveis pelo fato do computador apresentar limitações de armazenamento. O que se procurou fazer nesta dissertação, foi assegurar que a órbita verdadeira estivesse contida nas pseudo-órbitas obtidas com extensões intervalares da função de um mapa discreto. Isso fez com que fosse definida uma margem de erro para os resultados, fixando uma precisão para os cálculos realizados.

A metodologia desenvolvida consistiu na junção de dois métodos de análise intervalar. O primeiro método obtém respostas a cada iteração em uma função recursiva intervalar a partir de operações aritméticas computacionais dos limites do intervalo da iteração anterior. Trabalhando apenas com os limites do intervalo, se garante que todas as possíveis respostas dos valores per-

tencentes ao intervalo estejam limitadas entre as respostas destes limites. Além disso, ao não realizar operações aritméticas intervalares, obtém-se uma redução considerável na superestimação dos intervalos, problema que inviabiliza a análise intervalar em muitos casos.

O segundo método empregado foi uma tentativa de reduzir o tamanho dos intervalos obtidos por meio da interseção entre as respostas obtidas, a cada iteração, de diferentes extensões intervalares de uma função que representa um mapa. Foi evidenciado que, se a interseção é realizada ao final dos cálculos de pseudo-órbitas obtidas por extensões intervalares distintas, a partir dos mesmos parâmetros, a interseção será menor ou pelo menos do mesmo tamanho do menor intervalo das pseudo-órbitas a cada iteração, desde que a condição inicial pertença aos intervalos iniciais.

Porém, aplicar a interseção depois dos cálculos das pseudo-órbitas força que a resposta do método tenha o mesmo aspecto das pseudo-órbitas, caso estas convirjam para órbitas periódicas de mesmo período. Isso faria com que apenas os tamanhos dos intervalos obtidos fossem reduzidos ou pelo menos iguais ao menor dos intervalos das pseudo-órbitas a cada iteração.

Por isso, o método implementado foi uma adaptação do que foi provado ao longo do trabalho, com a intenção de deixar as respostas obtidas mais independentes das respostas apresentadas pelas extensões intervalares: a interseção é feita a cada saída obtida na iteração, e o intervalo produzido representa o intervalo da pseudo-órbita do método, servindo como variável de entrada para os cálculos com as diferentes extensões na próxima iteração.

Foram analisados 12 casos de parâmetros e condições iniciais do mapa logístico que, segundo Rasband (2015) e Feigenbaum (1978), deveriam convergir para órbitas de períodos 2^{n-1}, com $n \in \mathbb{N}$ representando o caso estudado. Os métodos desenvolvidos nesta dissertação foram comparados com o método intervalar desenvolvido por Nepomuceno (2014) e com o cálculo dos intervalos formados a partir da condição inicial com o *software Intlab*.

Para os quatro primeiros casos abordados, todos os métodos foram eficazes em atingir as órbitas periódicas esperadas de acordo com a literatura. Para o quinto caso, o método intervalar usando *Intlab* não foi capaz de obter a resposta esperada, convergindo para uma órbita com o período 8, ao invés de 16. A partir do sexto caso, tanto o método usando o *software* quanto o método de Nepomuceno (2014) não atingiram órbitas com os períodos especificados pela literatura, fixando suas respostas em período 8 para o *Intlab* e 16 para Nepomuceno (2014). É importante levar em conta que a precisão associada a essas respostas equivocadas é bem baixa,

uma vez que os tamanhos dos intervalos obtidos não são pequenos.

Assim, o método usando *Intlab* não alcançou períodos maiores que 8 e o método de Nepomuceno (2014) não obteve êxito no cálculo de órbitas com períodos maiores que 16. Isso ocorreu porque a ferramenta computacional realiza operações aritméticas intervalares e o método desenvolvido por Nepomuceno (2014) considera erros computacionais com certa taxa de confiança, de modo a possibilitar sua aplicação em outros mapas. Esses fatores são a principal influência no rápido crescimento dos intervalos a cada iteração e, consequentemente, fizeram com que os métodos não fossem capazes de distinguir órbitas com períodos superiores.

Nota-se que os tamanhos dos intervalos obtidos pela metodologia desenvolvida neste trabalho são pequenos em relação ao conjunto que representa a faixa de valores em que a variável x do mapa poderia assumir, onde $0 < x < 1$. Isso faz com que, mesmo necessitando de muito mais iterações em relação aos outros métodos, a precisão dos resultados obtidos seja satisfatória para confirmar que a metodologia é uma ferramenta a ser considerada na análise de órbitas no mapa logístico.

Para se ter uma ideia, no Caso 4 (Tabela 5.4), onde todos os métodos apresentam resultados de acordo com a literatura, o número de iterações necessárias para a convergência do método desenvolvido $F_1 \cap F_2$ foi mais de 8 vezes o número de iterações do método de Nepomuceno (2014) e mais de 13 vezes em relação ao *Intlab*. No entanto, o tamanho do intervalo final obtido com a metodologia desenvolvida foi de mais de 10^{12} vezes menor em relação às demais metodologias.

Ainda fazendo referência aos números de iterações necessários para a convergência em cada caso, a sugestão que fica é que a partir do momento em que o valor do parâmetro r aumenta, aproximando de regiões caóticas do mapa logístico e aumentando a inclinação do *cobweb plot*, mais iterações são necessárias para convergência das pseudo-órbitas e maiores os intervalos obtidos em todos os métodos.

Outro aspecto notado na avaliação dos resultados é que os intervalos obtidos como resposta no mapa logístico nem sempre crescem de uma iteração para outra. Isso depende da inclinação da curva do *cobweb plot*, definida pelo parâmetro r, e pela região em que o intervalo se encontra neste gráfico. Portanto, quanto mais próximo o intervalo estiver dos pontos limites do *cobweb plot* em uma iteração n, maior será a resposta desse intervalo, usando a metodologia desenvolvida, na iteração $n + 1$. Por outro lado, para intervalos localizados no centro desse

gráfico, podem ocorrer até reduções em seus tamanhos de uma iteração para outra.

Foi verificado que o método $F_1 \cap F_2$ nem sempre obteve intervalos finais menores em relação a F_1 e F_2 aplicados sob a mesma metodologia. Isso mostra que o método $F_1 \cap F_2$ não pode ser usado para reduzir o tamanho dos intervalos obtidos com F_1 e F_2 em todos os casos. Porém, os resultados obtidos com esse método foram melhores em outros aspectos, como na obtenção de valores médios e máximos de intervalos menores em relação a F_1 e F_2. Isso quer dizer que, de uma forma geral, no método que usa a interseção a cada iteração, os intervalos tendem a apresentarem tamanhos menores em relação aos demais métodos aplicados nessa metodologia.

Finalmente, enfatiza-se a eficácia da metodologia desenvolvida nesse trabalho, uma vez que foi a única a alcançar e garantir os resultados estabelecidos na literatura. Além disso, os tamanhos dos intervalos evidenciam as altas precisões das respostas obtidas.

6.2 Trabalhos Futuros

Espera-se que com o uso de mais extensões intervalares, os resultados possam ser aprimorados, o que é uma incumbência para investigações futuras.

Pretende-se expandir os conceitos e definições vistos ao longo dessa dissertação, aplicando a metodologia desenvolvida em outros mapas discretos unidimensionais. Além disso, também é pretendida a realização de adaptações nos métodos para que seja também possível aplicá-los em mapas com dimensões superiores.

Outra pretensão é investigar o comportamento do método em regime caótico e examinar como seu uso pode ser empregado na construção de diagramas de bifurcação do mapa logístico e outros mapas.

Referências Bibliográficas

Alefeld, G. e Claudio, D. (1998). The basic properties of interval arithmetic, its software realizations and some applications. *Computers and Structures*, 67(1-3):3–8.

Alefeld, G. e Herzberger, J. (2012). *Introduction to interval computation*. Academic press.

Alefeld, G. e Mayer, G. (2000). Interval analysis: Theory and applications. *Journal of Computational and Applied Mathematics*, 121(1):421–464.

Alpar, O. (2014). Analysis of a new simple one dimensional chaotic map. *Nonlinear Dynamics*, 78(2):771–778.

Banerjee, S., Halder, K., Dasgupta, S., Mukhopadhyay, S., Ghosh, K., e Gupta, A. (2015). An Interval Approach for Robust Control of a Large PHWR with PID Controllers. *IEEE Transactions on Nuclear Science*, 62(1):281–292.

Billings, S. A. (2013). *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons.

Brousentsov, N. P. (1960). Computing machine "Setun"of Moscow State University. *New developments on computer technology*, pp. 226–234.

Campos, M. A. e Lima, P. F. (2005). Introdução ao tratamento da informação nos ensinos fundamental e médio. In *Notas em Matemática Aplicada do CNMAC XXVI - Congresso Nacional de Matemática Aplicada e Computacional*, volume 16, pp. 66, São Carlos - SP. SBMAC.

Chaitin-Chatelin, F. (1996). Is finite precision arithmetic useful for physics? *Journal of Universal Computer Science*, 2(5):380–395.

Chua, L. O. e Lin, G.-N. (1990). Canonical realization of chua's circuit family. *IEEE transactions on Circuits and Systems*, 37(7):885–902.

de Oliveira, L. W., Seta, F. d. S., e de Oliveira, E. J. (2016). Optimal reconfiguration of distribution systems with representation of uncertainties through interval analysis. *International Journal of Electrical Power & Energy Systems*, 83:382–391.

Devaney, R. L. (1992). *A First Course in Chaotic Dynamical Systems : Theory and Experiment*. Westview Press.

Doucet, A., De, N., e Gordon, N. (2001). An introduction to sequential monte carlo methods. In *Sequential Monte Carlo methods in practice*, pp. 3–14. Springer.

Faranda, D., Mestre, M. F., e Turchetti, G. (2012). Analysis of round off errors with reversibility test as a dynamical indicator. *International Journal of Bifurcation and Chaos*, 22(9):1250215.

Feigenbaum, M. J. M. J. (1978). Quantitative universality for a class of nonlnear transformations. *Journal of Statistical Physics*, 19(1):25–52.

Freitas, C. J. (2002). The issue of numerical uncertainty. *Applied Mathematical Modelling*, 26(2):237–248.

Galias, Z. (2002). Proving the existence of long periodic orbits in 1D maps using interval Newton method and backward shooting. *Topology and its Applications*, 124(1):25–37.

Galias, Z. (2013). The Dangers of Rounding Errors for Simulations and Analysis of Nonlinear Circuits and Systems?and How to Avoid Them. *IEEE Circuits and Systems Magazine*, 13(3):35–52.

Gilmore, R. e Lefranc, M. (2012). *The topology of chaos: Alice in stretch and squeezeland*. John Wiley & Sons.

Goldberg, D. (1991). What every computer scientist should know about floating-point arithmetic. *Computing Surveys*, 23(1):5–48.

Grigoletti, P. S., Dimuro, G. P., e Barboza, L. V. (2006). Análise Intervalar de Circuitos Elétricos. *Tendências em Matemática Aplicada e Computacional*, 2(2):287–296.

Güémez, J. e A., M. M. (1993). Control of chaos in unidimensional maps. *Physics Letters A*, 181(1):29–32.

Hammel, S. M., Yorke, J. A., e Grebogi, C. (1987). Do numerical orbits of chaotic dynamical processes represent true orbits? *Journal of Complexity*, 3(2):136–145.

Hargreaves, G. I. (2002). Interval Analysis in Matlab. *Analysis*, 416(416):1–49.

Heath, M. T. (2002). *Scientific computing*. McGraw-Hill : New York.

Heath, T. L. (1897). *The works of Archimedes*. Cambridge University Press : Cambridge.

Hénon, M. (1976). A two-dimensional mapping with a strange attractor. *Communications in Mathematical Physics*, 50(1):69–77.

Hickey, T., Ju, Q., e Van Emden, M. H. (2001). Interval arithmetic: From principles to implementation. *Journal of the ACM (JACM)*, 48(5):1038–1068.

IEEE (2008). *Institute of Electrical and Electronics Engineers. 754-2008 – IEEE Standard for Floating-Point Arithmetic*. IEEE. p. 1–58.

IEEE (2015). *Institute of Electrical and Electronics Engineers. IEEE Standard for Interval Arithmetic*. IEEE.

Klatte, R., Kulisch, U., Neaga, M., Ratz, D., e Ullrich, C. (2012a). *PASCAL-XSC: language reference with examples*. Springer Science & Business Media.

Klatte, R., Kulisch, U., Wiethoff, A., e Rauch, M. (2012b). *C-XSC: A C++ class library for extended scientific computing*. Springer Science & Business Media.

Kulisch, U. W. (1999). Advanced arithmetic for the digital computer, design of arithmetic units. *Electronic Notes of Theoritical Computation Science*, 24:63.

Leeser, M., Mukherjee, S., Ramachandran, J., e Wahl, T. (2014). Make it real: Effective floating-point reasoning via exact arithmetic. *Design, Automation & Test in Europe Conference & Exhibition (DATE)*, pp. 1–4.

Lorenz, E. N. (1963). Deterministic nonperiodic flow. *Journal of the Atmospheric Sciences*, 20(2):130–141.

Lozi, R. (2013). Can we trust in numerical computations of chaotic solutions of dynamical systems? *Topology and Dynamics of Chaos*, pp. 63–98.

Marshall, E. (1992). Fatal error: how patriot overlooked a scud. *Science*, 255(5050):1347–1347.

May, R. M. (1976). Simple mathematical models with very complicated dynamics. *The Journal Nature*, (261):459–467.

Mendonça, A. F. e Campos, M. A. (2013). Confiabilidade Autovalidável para Sistemas com Processo de Falhas Exponencial. *Tendências em Matemática Aplicada e Computacional*, 14(3):383–398.

Monteiro, L. H. A. (2006). *Sistemas dinâmicos*. Editora Livraria da Física.

Moore, R. E. (1966). *Interval analysis*, volume 4. Prentice-Hall Englewood Cliffs.

Moore, R. E. e Bierbaum, F. (1979). *Methods and applications of interval analysis*, volume 2. SIAM.

Moore, R. E., Kearfott, R. B., e Cloud, M. J. (2009). *Introduction to Interval Analysis*, volume 22. Society for Industrial and Applied Mathematics : Philadelphia.

Nepomuceno, E. G. (2014). Convergence of recursive functions on computers. *The Journal of Engineering*, pp. 1–3.

Nepomuceno, E. G. e Martins, S. A. M. (2016). A lower-bound error for free-run simulation of the polynomial narmax. *Systems Science & Control Engineering*, (ja):1–14. Doi: 10.1080/21642583.2016.1163296.

Nepomuceno, E. G., Takahashi, R. H., Amaral, G. F., e Aguirre, L. A. (2003). Nonlinear identification using prior knowledge of fixed points: a multiobjective approach. *International Journal of Bifurcation and Chaos*, 13(05):1229–1246.

Ott, E. (2002). *Chaos in Dynamical Systems (Second edition)*. Cambridge University Press : Cambridge.

Overton, M. L. (2001). *Numerical Computing with IEEE Floating Point Arithmetic*. Filadélfia: Society for Industrial and Applied Mathematics.

Paiva, B. P. O., Nepomuceno, E. G., e Amaral, G. F. V. (2015). Considerações sobre a condição inicial na construção do diagrama de bifurcação para o mapa logístico. *Anais do DINCON 2015 - Conferência Brasileira de Dinâmica, Controle e Aplicações.*, pp. 1–8.

Paul, A., Das, N., e Prusty, A. K. (2013). An advanced gray image encryption scheme by using discrete logarithm with logistic and heh64 chaotic functions. In *Advance Computing Conference (IACC), 2013 IEEE 3rd International*, pp. 1114–1120. IEEE.

Pereira, L. E. S. e Da Costa, V. M. (2014). Interval analysis applied to the maximum loading point of electric power systems considering load data uncertainties. *International Journal of Electrical Power and Energy Systems*, 54:334–340.

Rasband, S. N. (2015). *Chaotic dynamics of nonlinear systems*. Courier Dover Publications : Nova Iorque.

Richardson, L. F. (1911). The approximate arithmetical solution by finite differences of physical problems involving differential equations, with an application to the stresses in a masonry dam. *Philosophical Transactions of the Royal Society of London. Series A, Containing Papers of a Mathematical or Physical Character*, 210:307–357.

Rodrigues Junior, H. M. e Nepomuceno, E. G. (2015). Uso da Computação Por Intervalos para Cálculo de Ponto Fixo de um Mapa Discreto. *Anais do DINCON 2015 - Conferência Brasileira de Dinâmica, Controle e Aplicações.*, pp. 1–7.

Rodrigues Junior, H. M., Peixoto, M. L., Nardo, L. G., Nazaré, T. E., e Nepomuceno, E. G. (2016a). Cálculo de Pontos Fixos por Análise Intervalar para Mapas Discretos. *Anais do CBA 2016 - XXI Congresso Brasileiro de Automática*, pp. 1–6.

Rodrigues Junior, H. M., Peixoto, M. L., e Nepomuceno, E. G. (2016b). Simulation of Recursive Functions by Means of Interval Analysis and Pseudo-Orbits. *Proceedings of 6th International Conference on Nonlinear Science and Complexity, NSC 2016.*, pp. 1–4.

Rothwell, E. J. e Cloud, M. J. (2012). Automatic error analysis using intervals. *IEEE Transactions on Education*, 55(1):9–15.

Rudolph-Lilith, M. e Muller, L. E. (2014). On a representation of the verhulst logistic map. *Discrete Mathematics*, 324:19–27.

Ruggiero, M. A. G. e Lopes, V. L. d. R. (1997). *Cálculo numérico: aspectos teóricos e computacionais*. Makron Books do Brasil.

Rump, S. M. (1999). INTLAB - INTerval LABoratory. In Csendes, T., editor, *Developments in Reliable Computing*, pp. 77–104. Kluwer Academic Publishers, Dordrecht.

Saha P. and Strogatz, S. H. (1995). The Birth of Period Three. *Mathematics Magazine*, 68(1):42–47.

Scheinerman, E. (2011). Invitation to Dynamical Systems. *Journal of Neural Engineering*, 8(6):065004.

Soares, G. L., Guimarães, F. G., Maia, C. A., Vasconcelos, J. A., e Jaulin, L. (2009). Interval robust multi-objective evolutionary algorithm. In *2009 IEEE Congress on Evolutionary Computation, CEC 2009*, pp. 1637–1643.

Spandl, C. (2012). Computational complexity of iterated maps on the interval. *Mathematics and Computers in Simulation*, 82(8):1459–1477.

Tomida, A. G. (2008). Matlab toolbox and GUI for analyzing one-dimensional chaotic maps. *Proceedings of International Conference on Computational Sciences and its Applications, ICCSA 2008*, pp. 321–330.

Tuinenga, P. W. (1995). *SPICE: a guide to circuit simulation and analysis using PSpice*. Prentice Hall PTR.

Viana, G. V. R. (1999). Padrão IEEE 754 para Aritmética Binária de Ponto Flutuante. *Revista CT*, 1(1):29–33.

Viswanath, D. (2000). Random fibonacci sequences and the number 1.13198824.... *Mathematics of Computation of the American Mathematical Society*, 69(231):1131–1155.

Walter, W. V. (1993a). ACRITH-XSC A Fortran-like Language for Verified Scientific Computing. *Mathematics in science and engineering*, 189:45–70.

Walter, W. V. (1993b). FORTRAN-XSC: A Portable Fortran 90 Module Library for Accurate and Reliable Scientific Computing. In *Validation Numerics*, pp. 265–285. Springer.

Weber, R. F. (2000). *Fundamentos de arquitetura de computadores*. Sagra Luzzatto.

Yabuki, M. e Tsuchiya, T. (2013). Double Precision Computation of the Logistic Map Depends on Computational Modes of the Floating-point Processing Unit. *ArXiv e-prints*, pp. 1–10.

APÊNDICE A

Rotinas Computacionais

A.1 Algoritmo da Aplicação de $F_1 \cap F_2$

```
%r = 330/100;     %Periodo 2
%r = 347/100;     %Periodo 4
%r = 355/100;     %Periodo 8
%r=3.566;         %Periodo 16
%r=3.5689;        %Periodo 32
%r = 3.5698;      %Periodo 64
%r = 3.5699;      %Periodo 128
%r = 3.56994;     %Periodo 256
%r = 3.569944;    %Periodo 512
%r = 3.5699453;   %Periodo 1024
%r = 3.5699456;   %Periodo 2048
N=25000;
system_dependent('setround',-Inf);  %Arredonda para baixo
x_inf(1)=0.6;
F1_inf(1)=0.6;
F2_inf(1)=0.6;
system_dependent('setround',Inf);   %Arredonda para cima
x_sup(1)=0.6;
F1_sup(1)=0.6;
F2_sup(1)=0.6;
system_dependent('setround',0.5);  %Arredonda para o mais proximo
E = x_sup-x_inf;     %Tamanho do intervalo
```

```
pos = 1;     %Posicao do intervalo em relacao ao cobweb plot
             %0 - Parte Crescente
             %1 - Parte Drescente
             %2 - Partes Crescente e Decrescente
h = 0;  %Variavel que informa a ocorrencia de orbitas periodicas
        %nas operacoes realizadas com analise intervalar.

for n=1:N
    %F1(x)=r.x.(1-x) e  F2(x)=r.(x.(1-x))

    %Caso o intervalo esteja na parte crescente do cobweb plot
    if pos == 0
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*x_inf(n)*(1-x_inf(n));
        F2_inf(n+1)=r*(x_inf(n)*(1-x_inf(n)));
        system_dependent('setround',Inf);
        F1_sup(n+1)=r*x_sup(n)*(1-x_sup(n));
        F2_sup(n+1)=r*(x_sup(n)*(1-x_sup(n)));
    %Caso o intervalo esteja na parte decrescente do cobweb plot
    elseif pos == 1
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*x_sup(n)*(1-x_sup(n));
        F2_inf(n+1)=r*(x_sup(n)*(1-x_sup(n)));
        system_dependent('setround',Inf);
        F1_sup(n+1)=r*x_inf(n)*(1-x_inf(n));
        F2_sup(n+1)=r*(x_inf(n)*(1-x_inf(n)));
    %Caso o intervalo esteja na parte crescente e decrescente do cobweb plot
    elseif pos == 2
        %Para o limite x mais distante de 0,5, temos que f(x)=min(f(inf),f(sup))
        if abs(0.5-x_inf(n)) < abs(0.5-x_sup(n))
        x_inf(n) = x_sup(n);
        end
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*x_inf(n)*(1-x_inf(n));
        F2_inf(n+1)=r*(x_inf(n)*(1-x_inf(n)));
        system_dependent('setround',Inf);
        F1_sup(n+1)=0.25*r;
        F2_sup(n+1)=0.25*r;
    end
```

```
    %Intersecao entre os intervalos obtidos por F1 e F2
    x_inf(n+1) = max(F1_inf(n+1),F2_inf(n+1));
    x_sup(n+1) = min(F1_sup(n+1),F2_sup(n+1));
    %Posicao do intervalo X(n+1)=[x_inf(n+1),x_sup(n+1)]
    if  x_sup(n+1) < 0.5
        pos = 0;
    elseif x_inf(n+1) > 0.5
        pos = 1;
    elseif x_sup(n+1) > 0.5 && x_inf(n+1) < 0.5
        pos = 2;
    end
    %Tamanho do Intervalo
    E(n+1) = x_sup(n+1)-x_inf(n+1);
    %------------Determinacao de pontos fixos
    for i=1:n
        if (n+1) >= (2*i)
            %Analisando intersecao entre intervalos
            if x_inf(n+1) <= x_sup(n+1-i) && x_inf(n+1-i) <= x_sup(n+1)
                h = 1;
                if i == 1
                    disp(['Ponto fixo.'])
                else
                    disp(['Orbita periodica de periodo ' num2str(i) '.'])
                    disp(['Iteracao: ' num2str(n)])
                    disp(' ')
                end
                disp(['Tamanho do intervalo final: ' num2str(E(n+1))]);
                break;
            end
        end
    end
    if h == 1
        break; %Em caso de ponto fixo ou orbita periodica, sai do segundo loop "for".
    end
end
```

A.2 Algoritmo do Cálculo de F_1 e F_2 Sem Usar Interseção das Pseudo-Órbitas

```
%r = 330/100;     %Periodo 2
%r = 347/100;     %Periodo 4
%r = 355/100;     %Periodo 8
%r=3.566;         %Periodo 16
%r=3.5689;        %Periodo 32
%r = 3.5698;      %Periodo 64
%r = 3.5699;      %Periodo 128
%r = 3.56994;     %Periodo 256
%r = 3.569944;    %Periodo 512
%r = 3.5699453;   %Periodo 1024
%r = 3.5699456;   %Periodo 2048
N=25000;

system_dependent('setround',-Inf);  %Arredonda para baixo
F1_inf(1)=0.6;
F2_inf(1)=0.6;

system_dependent('setround',Inf);   %Arredonda para cima
F1_sup(1)=0.6;
F2_sup(1)=0.6;

system_dependent('setround',0.5);   %Arredonda para o mais proximo
E1 = F1_sup-F1_inf;
E2 = F2_sup-F2_inf;

%Posicao do intervalo em relacao ao cobweb plot
pos_F1 = 1;
pos_F2 = 1;

%Variavel que informa a ocorrencia de orbitas periodicas
%nas operacoes realizadas com analise intervalar.
h1=0; h2=0;
```

```
for n=1:N

    %F1(x)=r.x.(1-x)

    %Caso o intervalo esteja na parte crescente do cobweb plot
    if pos_F1 == 0
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*F1_inf(n)*(1-F1_inf(n));
        system_dependent('setround',Inf);
        F1_sup(n+1)=r*F1_sup(n)*(1-F1_sup(n));
    %Caso o intervalo esteja na parte decrescente do cobweb plot
    elseif pos_F1 == 1
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*F1_sup(n)*(1-F1_sup(n));
        system_dependent('setround',Inf);
        F1_sup(n+1)=r*F1_inf(n)*(1-F1_inf(n));
    %Caso o intervalo esteja na parte crescente e decrescente do cobweb plot
    elseif pos_F1 == 2
        %Para o limite x mais distante de 0,5, temos que f(x)=min(f(inf),f(sup))
        if abs(0.5-F1_inf(n)) < abs(0.5-F1_sup(n))
            F1_inf(n) = F1_sup(n);
        end
        system_dependent('setround',-Inf);
        F1_inf(n+1)=r*F1_inf(n)*(1-F1_inf(n));
        system_dependent('setround',Inf);
        F1_sup(n+1)=0.25*r;
    end

    %Posicao do intervalo seguinte
    if  F1_sup(n+1) < 0.5
        pos_F1 = 0;
    elseif F1_inf(n+1) > 0.5
        pos_F1 = 1;
    elseif F1_sup(n+1) > 0.5 && F1_inf(n+1) < 0.5
        pos_F1 = 2;
    end

    %Tamanho do Intervalo de F1
    E1(n+1) = F1_sup(n+1)-F1_inf(n+1);
```

```
%F2(x)=r.(x.(1-x))

%Caso o intervalo esteja na parte crescente do cobweb plot
if pos_F2 == 0
    system_dependent('setround',-Inf);
    F2_inf(n+1)=r*(F2_inf(n)*(1-F2_inf(n)));
    system_dependent('setround',Inf);
    F2_sup(n+1)=r*(F2_sup(n)*(1-F2_sup(n)));
%Caso o intervalo esteja na parte decrescente do cobweb plot
elseif pos_F2 == 1
    system_dependent('setround',-Inf);
    F2_inf(n+1)=r*(F2_sup(n)*(1-F2_sup(n)));
    system_dependent('setround',Inf);
    F2_sup(n+1)=r*(F2_inf(n)*(1-F2_inf(n)));
%Caso o intervalo esteja na parte crescente e decrescente do cobweb plot
elseif pos_F2 == 2
    %Para o limite x mais distante de 0,5, temos que f(x)=min(f(inf),f(sup))
    if abs(0.5-F2_inf(n)) < abs(0.5-F2_sup(n))
        F2_inf(n) = F2_sup(n);
    end
    system_dependent('setround',-Inf);
    F2_inf(n+1)=r*(F2_inf(n)*(1-F2_inf(n)));
    system_dependent('setround',Inf);
    F2_sup(n+1)=0.25*r;
end

%Posicao do intervalo seguinte
if  F2_sup(n+1) < 0.5
    pos_F2 = 0;
elseif F2_inf(n+1) > 0.5
    pos_F2 = 1;
elseif F2_sup(n+1) > 0.5 && F2_inf(n+1) < 0.5
    pos_F2 = 2;
end

%Tamanho do intervalo
E2(n+1) = F2_sup(n+1)-F2_inf(n+1);
```

```
    %PONTOS FIXOS E ORBITAS PERIODICAS
    if h1 == 0
        for i=1:n
            if (n+1) >= (2*i)
                if F1_inf(n+1) <= F1_sup(n+1-i) && F1_inf(n+1-i) <= F1_sup(n+1)
                    h1 = 1;
                    if i == 1
                        disp(['Ponto fixo encontrado para F1'])
                    else
                        disp(['Orbita periodica de periodo ' num2str(i) para F1'])
                    end
                    disp(['O tamanho do intervalo final para F1:' num2str(E1(n+1))]);
                    break;
                end
            end
        end
    end

    if h2 == 0
        for i=1:n
            if (n+1) >= (2*i)
                    if F2_inf(n+1) <= F2_sup(n+1-i) && F2_inf(n+1-i) <= F2_sup(n+1)
                    h2 = 1;
                    disp(' ')
                    if i == 1
                        disp(['Ponto fixo encontrado para F2'])
                    else
                        disp(['Orbita periodica de periodo ' num2str(i) para F2'])
                    end
                    disp(['O tamanho do intervalo final para F2:' num2str(E2(n+1))]);
                    break;
                end
            end
        end
    end
    if h1 == 1 && h2 == 1
        break;
    end
end
```